Matching (Multi)Cut: Algorithms, Complexity, and Enumeration

Guilherme C. M. Gomes
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
CNRS/LIRMM, Montpellier, France

Emanuel Juliano
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Gabriel Martins
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Vinicius F. dos Santos
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

Abstract

A matching cut of a graph is a partition of its vertex set in two such that no vertex has more than one neighbor across the cut. The Matching Cut problem asks if a graph has a matching cut. This problem, and its generalization d-cut, has drawn considerable attention of the algorithms and complexity community in the last decade, becoming a canonical example for parameterized enumeration algorithms and kernelization. In this paper, we introduce and study a generalization of Matching Cut, which we have named Matching Multicut: can we partition the vertex set of a graph in at least ℓ parts such that no vertex has more than one neighbor outside its part? We investigate this question in several settings. We start by showing that, contrary to Matching Cut, it is NP-hard on cubic graphs but that, when ℓ is a parameter, it admits a quasi-linear kernel. We also show an $\mathcal{O}(\ell^{\frac{n}{2}})$ time exact exponential algorithm for general graphs and a $2^{\mathcal{O}(t \log t)} n^{\mathcal{O}(1)}$ time algorithm for graphs of treewidth at most t. We then turn our attention to parameterized enumeration aspects of matching multicuts. First, we generalize the quadratic kernel of Golovach et. al for Enum Matching Cut parameterized by vertex cover, then use it to design a quadratic kernel for Enum Matching (Multi)cut parameterized by vertex-deletion distance to co-cluster. Our final contributions are on the vertex-deletion distance to cluster parameterization, where we show an FPT-delay algorithm for Enum Matching Multicut but that no polynomial kernel exists unless NP $\subseteq$ coNP/poly; we highlight that we have no such lower bound for Enum Matching Cut and consider it our main open question.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Matching Cut, Matching Multicut, Enumeration, Parameterized Complexity, Exact exponential algorithms

Digital Object Identifier 10.4230/LIPIcs.IPEC.2024.25

Related Version *Full Version*: https://arxiv.org/abs/2407.02898

Funding *Guilherme C. M. Gomes*: Partially funded by the European Union, project PACKENUM, grant number 101109317. Views and opinions expressed are however those of the author only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.
Vinicius F. dos Santos: Partially funded by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), grants 312069/2021-9, 406036/2021-7 and 404479/2023-5 by Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG), grant APQ-01707-21.

19th International Symposium on Parameterized and Exact Computation (IPEC 2024).
Editors: Édouard Bonnet and Paweł Rzążewski; Article No. 25; pp. 25:1–25:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

A matching M in a graph G is a subset of the edges of G such that no vertex is the endpoint of more than one edge in M. Matchings are one of the most fundamental concepts in graph theory, with whole books dedicated to them [33, 34]. A cut of a graph G is a partition of its vertex set into two non-empty sets and we say that the set of edges between them is an edge cut. A matching cut is a edge cut that is also a matching. Not all graphs admit a matching cut, and graphs admitting such kind of cuts were first considered by Graham [28], who called them decomposable graphs, to solve a problem in number theory. Other applications include fault-tolerant networks [20], multiplexing networks [1] and graph drawing [41]. The problem of recognizing graphs that do admit a matching cut, called Matching Cut, was studied by Chvátal [13], who proved that the problem is NP-complete even restricted to graphs of maximum degree four, while polynomial-time solvable in graphs of maximum degree three. The problem was reintroduced under the current terminology in [41] and, since then, it has been attracting much attention of the algorithms community. It has also been shown to remain NP-complete for several graph classes such as bipartite graphs of bounded degree [42], planar graphs of bounded degree or bounded girth [8] and P_t-free graphs (for large enough t) [21]. On the positive side, tractable cases include H-free graphs, i.e. graphs without an induced subgraph isomorphic to H, for some H, including P_6, the path on 6 vertices [36]. For a more comprehensive overview and recent developments, we refer to [12, 38].

Matching Cut has also been studied from the parameterized perspective, with the minimum number of edges crossing the cut k being used as the natural parameter for this problem. The first parameterized algorithm for k was given by Marx et al. in [39]; they tackled the Stable Cutset problem using the treewidth reduction machinery and Courcelle's theorem, which yielded a very large dependency on k. We remark that Matching Cut on G is equivalent to finding a separator that is an independent set in the line graph of G. Using the compact tree decomposition framework of Cygan et al. [16], Aravind and Saxena [4] developed a $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ time algorithm for Matching Cut. Komusiewicz et al. [31] presented a quadratic kernel for the vertex-deletion distance to cluster parameterization, as well as single exponential time FPT algorithms for this parameterization and for vertex-deletion distance to co-cluster; on the other hand, they gave a kernelization lower bound for the combined parameterization of treewidth plus the number of edges in the cut. Aravind et al. [3] presented FPT algorithms for neighborhood diversity, twin-cover and treewidth for Matching Cut; the latter had its running time improved by Gomes and Sau in [26].

One area in which matching cuts have drawn particular attention is in parameterized enumeration. Under this framework, our goal is to list all feasible solutions to a problem, e.g. all matching cuts of an input graph. Parameterized algorithms that do so are classified in two families: TotalFPT – where all solutions can be listed in FPT time – and DelayFPT – where the delay between outputting two solutions, i.e. the time between these outputs, is at most FPT. Based on the foundational work of Creignou et al. [14], Golovach et al. [25] defined the kernelization analogues of TotalFPT and DelayFPT. Also in [25], the authors developed several enumeration and kernelization algorithms for Enum Matching Cut under the vertex cover, neighborhood diversity, modular width, and clique partition number parameterizations. They also studied the enumeration of minimal and maximal matching cuts in the form of the Enum Minimal MC and Enum Maximal MC problems under some of the aforementioned parameterizations.

Similar problems to Matching Cut, as well as minimization and maximization questions [37], have also been considered. Their hardness follow directly from the problem definition. Another related problem, Perfect Matching Cut, asks for the existence of a perfect matching that is also a matching cut. Although its hardness does not follow directly from Matching Cut the problem is also NP-complete [30]. The recent survey by Le et al. [32] revisit and compare results on these variations. Some problems, however, can be seen as direct generalizations of Matching Cut. In the d-Cut problem, the goal is to partition the vertex set into two sets such that each vertex has at most d neighbors in the opposite set of the partition. Introduced in [26], d-Cut has been shown to be NP-complete for $(2d + 2)$-regular graphs and it has been shown to admit FPT algorithms under several parameters such as the maximum number of edges crossing the cut [4], treewidth, vertex-deletion distance to cluster, and vertex-deletion distance to co-cluster. When $d = 1$ the problem is exactly Matching Cut. However, many cases that are tractable for $d = 1$ have been shown to become hard for d-Cut [35]. The other related problem arises in the context of graph convexity. To our purposes, a convexity is a family $\mathcal{C}$ of subsets of a finite ground set X such that $X, \emptyset \in \mathcal{C}$ and $\mathcal{C}$ is closed for intersection. Many graph convexities have been considered in the literature [2, 11, 17, 29], most of them motivated by families of paths. In this context, a subset S of vertices is convex if all paths of a given type between vertices of S contain only vertices of S. The most well-studied paths in the literature are shortest paths, induced paths and P_3, the paths on three vertices. One of the problems studied in the graph convexity setting is the partition of the vertex set of a graph into convex sets. Note that, in the P_3-convexity, this is equivalent to partition vertices in such a way that two vertices in a set S have no common neighbor outside S. Hence, partitioning into two P_3-convex sets is equivalent to Matching Cut. The more general case has also been considered in [10, 27].

Our contributions. In this work we introduce the Matching Multicut problem, a novel generalization of Matching Cut. A *matching multicut on ℓ parts* of a graph G is a partition of its vertex set in $\{A_1, \ldots, A_\ell\}$ such that each vertex in A_i has at most one neighbor outside of A_i. Note that this is quite different from a partition into P_3-convex sets; in the latter, a vertex $v \in A_i$ may have one neighbor in *each* other A_j, while in the former, v may have one neighbor in $\bigcup_{j \neq i} A_j$. Formally, we study the following problem:

Matching Multicut
Instance: A graph G and an integer ℓ.
Question: Does G admit a matching multicut on at least ℓ parts?

We explore the complexity landscape of Matching Multicut under several settings that were previously considered for Matching Cut. Since the case $\ell = 2$ is exactly Matching Cut, the problem is trivially NP-hard. It is also trivially paraNP-hard for the natural parameter ℓ. We study its complexity for cubic graphs, exact exponential algorithms, structural parameterizations as well parameterized enumeration questions.

Contrary to the classic result of Chvátal showing the polynomial-time solvability of Matching Cut [13] for cubic graphs, we show that Matching Multicut is NP-hard even restricted to those graphs. On the other hand, the problem becomes fixed parameter tractable when parameterized by ℓ. Indeed, we show that the problem admits a quasi-linear kernel under this parameterization for subcubic graphs. We also show that the problem is FPT parameterized by treewidth. From the definition of the problem, there is a trivial $\ell^n n^{\mathcal{O}(1)}$ time algorithm for Matching Multicut by just enumerating all possible (ordered) partitions of $V(G)$. We improve this by showing that the problem can be solved in $\alpha_\ell^n n^{\mathcal{O}(1)}$ time, with $\alpha_\ell \leq \sqrt{\ell}$ for a graph on n vertices. Finally, we turn our attention to the enumeration of matching multicuts in the form of the Enum Matching Multicut problem.

Enum Matching Multicut
Instance: A graph G and an integer ℓ.
Enumerate: All matching multicuts of G on at least ℓ parts

Our first results in this direction are a polynomial-delay enumeration (PDE) kernel under vertex cover and a PDE kernel under vertex-deletion distance to co-cluster. Afterwards, we present a DelayFPT algorithm for enumerating matching multicuts of a graph parameterized by the vertex-deletion distance to cluster. For our final result, we show that, although Enum Matching Multicut is in DelayFPT for the vertex-deletion distance to cluster parameter, Matching Multicut does not admit a polynomial kernel under the joint parameterization of distance to cluster, maximum cluster size and the number of parts of the cut, unless NP $\subseteq$ coNP/poly. To prove this result, we show that Set Packing has no polynomial kernel parameterized by the size of the ground set, which could be of independent interest. To the best of our knowledge, although expected, this result has not been explicitly stated before.

1.1 Preliminaries

We denote $\{1, 2, \dots, n\}$ by $[n]$. We say that a monotonically non-decreasing function f is *quasi-linear* if $f(n) \in \mathcal{O}(n \log^c n)$ for some constant c. We use standard graph-theoretic notation, and we consider simple undirected graphs without loops or multiple edges; see [6] for any undefined terminology. When the graph is clear from the context, the degree (that is, the number of neighbors) of a vertex v is denoted by $\deg(v)$, and the number of neighbors of a vertex v in a set $A \subseteq V(G)$ and its neighborhood in it are denoted by $\deg_A(v)$ and $N_A(v)$; we also define $N(S) = \bigcup_{v \in S} N(v) \setminus S$. The minimum degree and the maximum degree of a graph G are denoted by $\delta(G)$ and $\Delta(G)$, respectively. We say that G is cubic if $\deg(v) = 3$ for all $v \in V(G)$ and that G is subcubic if $\deg(v) \leq 3$. A *matching* M of G is a subset of edges of G such that no vertex of G is incident to more than one edge in M; for simplicity, we define $V(M) = \bigcup_{uv \in M} \{u, v\}$ and refer to it as the set of *M-saturated vertices*. The *subgraph of G induced by X* is defined as $G[X] = (X, \{uv \in E(G) \mid u, v \in X\})$. The vertex-deletion distance to $\mathcal{G}$ is the size of a minimum cardinality set $U \subseteq V(G)$ such that $G \setminus U = G[V(G) \setminus U]$ belongs to class $\mathcal{G}$; in this case, U is called the $\mathcal{G}$-modulator. A graph G is a *cluster graph* if each connected component is a clique; G is a co-cluster graph if its complement is a cluster graph. A *vertex cover* of G is a set of vertices incident to every edge of G. A tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of a connected graph G is such that T is a tree, $X_t \subseteq V(G)$ for all t and: (i) for every $uv \in E(G)$ there is some $t \in T$ where $u, v \in X_t$ and (ii) the nodes of T that contain $v \in V(G)$ form a subtree of T, for every v. The sets X_t are called the *bags* of the decomposition, the *width* of the decomposition is $\max_{t \in V(T)} X_t - 1$. The *treewidth* of G is the size of a tree decomposition of G of minimum width. For more on treewidth and, in particular, nice tree decompositions, we refer the reader to [15].

We refer the reader to [15, 18] for basic background on parameterized complexity, and we recall here only some basic definitions. A *parameterized problem* is a tuple (L, κ) where $L \subseteq \Sigma^*$ is a language and $\kappa : \Sigma^* \mapsto \mathbb{N}$ is a parameterization. For an instance $I = (x, k) \in \Sigma^* \times \mathbb{N}$, k is called the *parameter*. A parameterized problem is *fixed-parameter tractable* FPT if there exists an algorithm $\mathcal{A}$, a computable function f, and a constant c such that given an instance $I = (x, k)$, $\mathcal{A}$ (called an FPT *algorithm*) correctly decides whether $I \in L$ in time bounded by $f(k) \cdot |I|^c$. A fundamental concept in parameterized complexity is that of *kernelization*; see [23] for a recent book on the topic. A kernelization algorithm, or just *kernel*, for a parameterized problem Π takes an instance (x, k) of the problem and, in time polynomial in $|x| + k$, outputs an instance (x', k') such that $|x'|, k' \leqslant g(k)$ for some function g, and

$(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$. The function g is called the *size* of the kernel. A kernel is called *polynomial* (resp. *quadratic, linear*) if the function $g(k)$ is a polynomial (resp. quadratic, linear) function in k.

In terms of parameterized enumeration, we refer the reader to [14, 25] for a more comprehensive overview than what we give below. A *parameterized enumeration problem* is a triple $(L, \mathsf{Sol}, \kappa)$ where $L \subseteq \Sigma^*$ is a language, $\mathsf{Sol} : \Sigma^+ \mapsto 2^{\Sigma^*}$ is the set of all viable solutions and $\kappa : \Sigma^* \mapsto \mathbb{N}$ is a parameterization. An instance to a parameterized enumeration problem is a pair (x, k) where $k = \kappa(x)$ and the goal is to produce $\mathsf{Sol}(x)$. We say that an algorithm $\mathcal{A}$ that takes (x, k) as input is a TotalFPT algorithm if it outputs $\mathsf{Sol}(x)$ in FPT time. Naturally, several problems won't have $\mathsf{Sol}(x)$ of FPT size. In this case, the best we can hope for is that the *delay* to outputting a new solution is FPT. If not only this is the case but also: (i) the time to the first solution, and (ii) the time from the final solution to the halting of the algorithm are also in FPT, then we say that the algorithm is a DelayFPT algorithm. Very recently, Golovach et al. [25] gave kernelization analogues to TotalFPT and DelayFPT, which they called *fully-polynomial enumeration kernel* (FPE) and *polynomial-delay enumeration kernel* (PDE), respectively. Formally, an FPE kernel is a pair of algorithms $\mathcal{A}, \mathcal{A}'$ called the *compressor*[1] and *lifting* algorithms, respectively, where:

- Given (x, k), $\mathcal{A}$ outputs (x', k') with $|x'|, k' \leq g(k)$ in time $\mathsf{poly}(|x| + k)$ for some computable g.
- For each $s \in \mathsf{Sol}(x')$, $\mathcal{A}'$ computes a set S_s in time $\mathsf{poly}(|x| + |x'| + k + k')$ such that $\{S_s \mid s \in \mathsf{Sol}(x')\}$ is a partition of $\mathsf{Sol}(x)$.

For PDE kernels, we replace the polynomial (total) time condition of $\mathcal{A}'$ with *polynomial delay* on $|x| + |x'| + k + k'$.

2 (Sub)Cubic graphs

A result of Chvátal [13] from the 1980s shows that Matching Cut is polynomial-time solvable for subcubic graphs. Later, Moshi [40] showed that every connected subcubic graph on at least eight vertices has a matching cut. When dealing with Matching Multicut, the situation is not as simple. We first show that, if the number of components ℓ is part of the input, then Matching Multicut is NP-hard. However, we are able to prove a Moshi-like result, and show that, if ℓ is a parameter, then the problem admits a quasi-linear kernel.

2.1 NP-hardness

First, let us show a lemma and some helpful definitions for our construction.

▶ **Definition 1.** *A graph G is indivisible if and only if G has no matching cut. A set of vertices $X \subseteq V(G)$ is said to be indivisible if the subgraph of G induced by X is indivisible.*

We remark that the above definition is a conservative notion of togetherness; i.e. we do not require that X is together in every matching cut of G, we require it to be together *regardless* of the remainder of the graph that contains it.

▶ **Definition 2.** *Let $X \subset V(G)$ induce a connected subgraph of G with exactly one $u \in X$ such that $N(u) \setminus X \neq \emptyset$. If $|N(u) \setminus X| = 1$ we say $G[X]$ is a pendant subgraph of G and that X induces a pendant subgraph of G.*

[1] This was named the *kernelization* algorithm in [25], but we reserve this term to the pair $\mathcal{A}, \mathcal{A}'$ itself.

▶ **Lemma 3.** *Let $I = \{H_1, \ldots, H_k\}$ be a set of maximal indivisible pendant subgraphs of G. Let $v_1, \ldots, v_k$ be pairwise distinct vertices so that $N(H_i) = \{v_i\}$. If (G, ℓ) is a yes-instance for some $\ell > k$, then there exists a matching multicut $\mathcal{P} = \{P_1, \ldots, P_\ell\}$ with $P_i = V(H_i)$ for $1 \leq i \leq k$.*

Construction. To construct our instance (H, ℓ) of Matching Multicut, we will reduce from an instance (G, k) of Independent Set on cubic graphs, which is a well known NP-complete problem [24]. First, for each $u \in V(G)$, create a K_3 in H, label it as B_u, and let $B = \bigcup_{u \in V(G)} B_u$. Suppose that $E(G)$ has been arbitrarily ordered as $\{e_1, \ldots, e_m\}$. For each edge in order, we add the gadget in Figure 1. The vertices in B_u and B_v are connected in such a way no vertex has more than three neighbors.

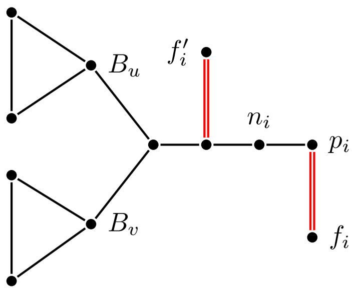

Figure 1 Edge gadget for edge $e_i = uv$. Thick edges are assumed to be in any solution to Matching Multicut.

To connect our edge edges, we add an edge between n_i and p_{i+1} for every $i \in [m-1]$ and between n_m and p_1. With this the subgraph of H induced by the n_i's and p_i'is a cycle on $2m$ vertices. Finally, we set $\ell = 2m + k + 1$. Intuitively, the pendant vertices force that the entire cycle and the vertices between the triangles are contained in a single part of the multicut; as such, picking a triangle invalidates picking any other triangle at distance two.

▶ **Lemma 4.** *$(H, 2|E(G)| + k + 1)$ is a* YES*-instance of Matching Multicut if and only if G has an independent set of size at least k. Moreover, Matching Multicut is* NP-complete *in subcubic graphs.*

We can show in a very similar manner that Matching Multicut is NP-hard for cubic graphs. To do this, we replace the pendant vertices of H with the indivisible graph in Figure 2. The remainder of the argument follows as in the proof of the previous theorem.

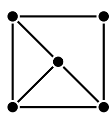

Figure 2 Indivisible pendant subgraph.

▶ **Corollary 5.** *Matching Multicut is* NP-complete *in cubic graphs.*

2.2 Quasi-linear kernel

We now present a quasi-linear kernel for the Matching Multicut problem in the case where the number of partitions ℓ is a parameter. In order to construct the kernel, we extend Moshi's result [40] and show the following theorem.

▶ **Theorem 6.** *Let G be a connected graph with $\Delta(G) \leq 3$. If $|V(G)| = \Omega(\ell \log^2 \ell)$, then G has a matching multicut that partitions the graph into ℓ parts.*

The key for proving Theorem 6 is to find a sufficiently large collection of vertex-disjoint cycles and construct a partition using some of them. In order to find these cycles, we first need to deal with vertices of degree at most 2.

▶ **Lemma 7.** *Let G be a connected subcubic graph, and let V_1 denote the vertices of G with degree 1. If $|V_1| \geq 3\ell$, then G has a matching multicut that partitions the graph into ℓ parts.*

Proof. Construct a matching M by greedily choosing edges that contain a vertex from V_1. Because $\Delta(G) \leq 3$, $|M| \geq \frac{1}{3}|V_1| \geq \ell$. Moreover, $G - M$ contains at least $|M|$ components with an isolated vertex. ◀

▶ **Lemma 8.** *Let G be a connected subcubic graph with $|V(G)| = \Omega(\ell)$, and let V_2 denote the vertices of G with degree 2. If $|V_2| \geq \frac{9}{10}|V(G)|$, then G has a matching multicut that partitions the graph into ℓ parts.*

Assume that G is a subcubic graph that does not satisfy the degree conditions of Lemmas 7 and 8. That is, G does not have many degree-1 vertices nor a large proportion of degree-2 vertices. Our strategy to find a matching multicut for G involves using disjoint cycles. The intuition here is that if a cycle C is entirely contained within a part of a partition of G, each vertex $v \in C$ will have at most one edge crossing the partition. Therefore, cycles are a good starting point for partitioning the matching multicut. However, first we need to find disjoint cycles. For this purpose, we utilize a theorem due to Simonovits [43], which is a precise version of the well-known Erdős–Pósa theorem [19] in the context of subcubic graphs.

▶ **Theorem 9** (Simonovits '67)**.** *Let G be a connected graph with $\delta(G) \geq 2$. Let $V_{\geq 3}$ be the set of vertices of G with degree at least 3. Then, G has at least $|V_{\geq 3}|/(4 \log |V_{\geq 3}|)$ vertex disjoint cycles.*

It is worth mentioning that there exists an algorithmic approximation of Theorem 9 due to Brandstädt and Voss [9]. Therefore, all the theorems presented in this subsection are constructive and can be used to find a matching multicut of a subcubic graph.

Before moving to the main theorem of this subsection, we make some observations about the neighborhood of subcubic graphs. Let $N(v)$ be the set of vertices of G adjacent to v, and let $N[v] = N(v) \cup \{v\}$. More generally, let $N(S)$ be the set of vertices of $G - S$ that are adjacent to some vertex in S, and let $N[S] = N(S) \cup S$. We call $N(S)$ the open neighborhood and $N[S]$ the closed neighborhood. We denote by $N^2[S] := N[N[S]]$ the closed square neighborhood. Notice that for subcubic graphs, $|N^2[S]| \leq 10|S|$.

Proof of Theorem 6. Let G be a graph satisfying the conditions of Theorem 6. Let $V_1(G)$, $V_2(G)$, and $V_3(G)$ be the subsets of vertices of G with degrees 1, 2, and 3, respectively. If G satisfies the conditions of Lemmas 7 or 8, we are done. Now we can safely assume that $|V_1(G)| < 3\ell$ and $|V_2(G)| < \frac{9}{10}|V(G)|$.

Let G' be the graph obtained from G after recursively removing degree 1 vertices. Notice that each time a degree 1 vertex is removed, a vertex moves from V_2 to V_1 or from V_3 to V_2. In both cases, the difference $|V_3| - |V_1|$ remains invariant; therefore, $|V_3(G')| \geq |V_3(G)| - |V_1(G)|$. Notice that any matching multicut of G' is also a matching multicut of G. Assume again that G' does not satisfy Lemma 8; in particular, this implies that $|V_3(G)| \geq |V(G)|/10$.

Now, G' satisfies the conditions of Theorem 9. Let $\{C_1, \ldots, C_k\}$ be a collection of $k = |V_3(G')|/(4 \log |V_3(G')|) = \Omega(\ell \log \ell)$ vertex-disjoint sets such that $G[C_i]$ is a cycle. By giving a lower bound for the value of k, we also give a lower bound for $|V(G)|$. Later in the proof, we will need k such that $k^2 \geq c\ell|V(G')|$, but notice that there always exists a constant $c' > c$ such that if $|V(G)| \geq c'\ell \log^2 \ell$, the lower bound on k^2 is satisfied.

For each set of vertices C_i, if there is $v \in V(G') \setminus C_i$ with $|N(v) \cap C_i| \geq 2$, add v to C_i, that is, $C_i := C_i \cup \{v\}$. Notice that with this process, every vertex inside C_i has at least two neighbors inside C_i, therefore, $E(C_i, V(G') \setminus C_i)$ forms a matching cut.

We construct the matching multicut greedily. Let $M := \emptyset$ be the initial matching multicut and let $S := \emptyset$ be a collection of marked vertices. Assume that the sets C_i are ordered by size with $|C_1| \leq \cdots \leq |C_k|$. Let C_i be a set in the first half of this ordering with no vertex marked, i.e., $C_i \cap S = \emptyset$. Add the edges with exactly one endpoint in C_i to M and mark $N^2[C_i]$, that is, $M := M \cup E(C_i, V(G') \setminus C_i)$ and $S := S \cup N^2[C_i]$. If no such C_i exists in the first half of the ordering, stop the process. We claim that in the end, M is indeed a matching multicut.

▶ **Lemma 10.** *If M is a set of edges constructed as above, then M is a matching multicut that divides G' into at least ℓ parts.*

It is easy to see that M is indeed a matching. Assuming otherwise, then there is a vertex v with two edges from M containing v. By previous observations, v must not belong to any set C_i whose border was added to M, thus $v \in V(G') \setminus (C_1 \cup \cdots \cup C_k)$. If $|N(v) \cap C_i| \geq 2$, v would already have been added to C_i, so this cannot be the case. Hence, there are distinct sets C_i and C_j chosen in the algorithm with $|N(v) \cap C_i|, |N(v) \cap C_j| \geq 1$. Assume that C_i was chosen before C_j. As v is adjacent to a vertex of C_i and a vertex of C_j, there is a vertex of C_j in $N^2[C_i]$, which means that this vertex should have been marked, implying that this situation also cannot happen. We conclude that there is no vertex v with two edges from M containing v.

Now, we just need to check that during the process at least ℓ sets C_i were chosen so that the edges $E(C_i, V(G') \setminus C_i)$ were added to M. If this does not occur, we have that the size of the marked vertices $|S|$ is bounded:

$$|S| \leq (\ell - 1) \max_{i \leq k/2} \{|N^2[C_i]|\} \leq 10(\ell - 1)|C_{k/2}| \leq 10(\ell - 1)\frac{|V(G)|}{k/2} < \frac{k}{2}$$

In the first inequality, we are assuming the worst case where we have always added the largest squared neighbourhood. The second inequality follows from our previous bound on the size of this squared closed neighbourhood. The third inequality holds because the average size of the $k/2$ largest sets C_i is $2|V(G)|/k$, and the set $C_{k/2}$ has a size below this average. The last inequality follows from our lower bound on k. This concludes the proof by showing that M indeed divides G' into at least ℓ components, so it is a matching multicut. ◀

Notice that in the proof, we choose $|V(G)|$ in order to establish a lower bound on k^2. We do not explicitly specify the choice of the constant c' such that $|V(G)| \geq c'\ell \log^2 \ell$. However, through a simple computation, it can be shown that $c' = 10^6$ is sufficient. We have not attempted to minimize the constants, but we believe that the value of c' can be reduced.

It follows from Theorem 6 that if we want to ask for a matching multicut that divides an n-vertex subcubic graph into $\ell = \mathcal{O}(n/\log^2 n)$ parts, the answer is trivially yes. On the other hand, Theorem 4 provides a construction of a subcubic graph and shows that it is NP-hard to determine if this graph has a matching multicut that divides it into $\Theta(n)$ parts. We leave it as an open question if it is possible to improve the asymptotic bound given by Theorem 6.

3 Exact Exponential Algorithm

We now turn our attention to developing an exact exponential algorithm through a similar approach used in [31]. For more on this type of algorithm and its associated terminology, we refer the reader to [22]. Our algorithm consists of four stopping rules, seven reduction and nine branching rules. At every step of the algorithm we have the sets $\{A_1, \ldots, A_\ell, F\}$ such that $\varphi = \{A_1, \ldots, A_\ell\}$ (unless any stopping rule is applicable) is a matching ℓ-multicut of the vertices of $V(G) \setminus F$. We set the size of the instance as the size of the set F, that is, how many free vertices are not assigned to any part yet. For simplicity, we assume that $\delta(G) \geq 2$. The arguments we use work with slight modifications to graphs of minimum degree one, but they would unnecessarily complicate the description of the algorithm.

Intuitively, stopping rules are applicable whenever a bad decision has been made by the branching algorithm and we must prune that branch. Reduction rules, on the other hand, are useful for cleaning up an instance after a branching step has been performed. Finally, our branching rules attempt to reduce the size of F as much as possible for each possibility. We follow the configurations given by Figure 3, and always branch on vertex v_1. The main culprit behind our complexity is rule B8, which gives us a branching vector of the form $\{1\} \times \{3\}^{\ell-1}$ and branching factor $\sqrt[3]{\ell} \leq \alpha_\ell \leq \sqrt{\ell}$ which, for $\ell >> 2$, will be our worst factor.

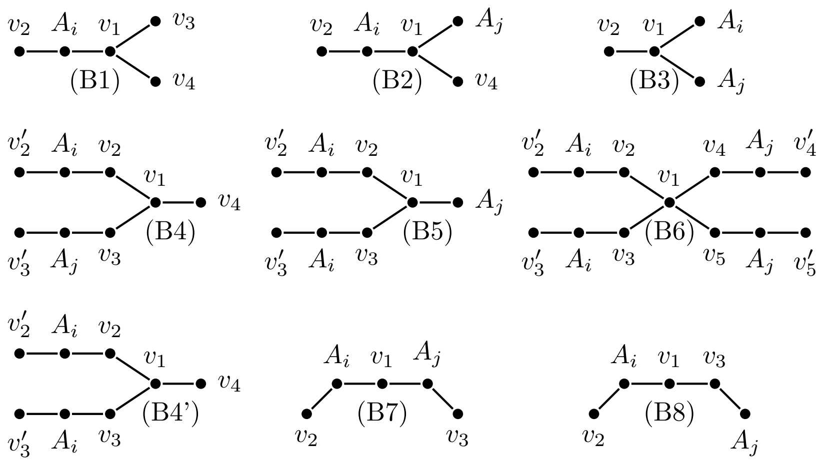

Figure 3 Branching configurations for MATCHING MULTICUT.

▶ **Theorem 11.** *MATCHING MULTICUT can be solved in $\alpha_\ell^n n^{\mathcal{O}(1)}$ time for a graph on n vertices, where $\alpha_\ell \leq \sqrt{\ell}$.*

4 FPT Algorithm by Treewidth

Let $(T, \{X_t\}_{t \in V(T)})$ be a nice tree decomposition of a graph G with n vertices, with T corresponding to the tree of the nice tree decomposition and X_t being the bag corresponding to vertex t. Suppose T is rooted at a vertex root, that $X_{\mathsf{root}} = \emptyset$. Let V_t be the union of all the bags present in the subtree rooted at t. Finally, define $G_t = G[V_t]$.

Our goal is to have $c[t, \mathcal{P}, \mathsf{Ext}] = \ell$ if and only if ℓ is the maximum integer such that (G_t, ℓ) is a YES instance of MATCHING MULTICUT that respects $\mathcal{P}$ and Ext, which we now formally define. First, $\mathcal{P}$ is a function $\mathcal{P} : X_t \mapsto X_t$ with $\mathcal{P}(v)$ corresponding to the vertex in X_t with the smallest label that is present in the same set as v in the partition. In other words, $\mathcal{P}$ is responsible for representing which partition of V_t we have assigned each vertex of X_t to. Note that $\mathcal{P}(v) \leq v$. Finally, $\mathsf{Ext} : X_t \longrightarrow \{0, 1\}$ is a function that signals whether each vertex in X_t has a neighbor in a different set in the partition. We denote by $\mathcal{P}|_{\overline{v}}$ the restriction of $\mathcal{P}$ to $X_t \setminus \{v\}$. Note that we can easily update each value in $\mathcal{P}|_{\overline{v}}$ to account for the missing vertex: we pick the minimum element in $\mathcal{P}^{-1}(v) \setminus \{v\}$ and set it as the new root of the component previously identified by v.

▶ **Theorem 12.** *If given a nice tree decomposition of width k of the n-vertex graph G, there exists an algorithm that solves* Matching MultiCut *in $2^{k \log k} n$ time.*

5 Matching Multicut Enumeration

5.1 Vertex Cover

In this section, we consider the parameterization of the matching multicut problem by the vertex cover number $\tau(G)$ of the input graph. This parameterization of Enum Matching Cut was previously studied in [25]. We show that the enumeration kernel constructed by the authors of [25] is also an enumeration kernel for Enum Matching Multicut. We assume that the vertex cover X of size $k \leq 2\tau(G)$ is given together with the input graph.

We describe the kernel constructed in [25]. Assume for simplicity that G contains no isolated vertices. Let $I = V(G) \setminus X$. Recall that I is an independent set. Denote by I_1 and $I_{\geq 2}$ the subsets of vertices of I with degree 1 and at least 2, respectively. We use the following marking procedure to label some vertices of I.

(i) For every $x \in X$, mark an arbitrary vertex of $N(x) \cap I_1$ (if it exists).
(ii) For every two distinct vertices $x, y \in X$, select an arbitrary set of $\min\{3, |N(x) \cap N(y) \cap I_{\geq 2}|\}$ vertices in $I_{\geq 2}$ adjacent to both x and y, and mark them for the pair $\{x, y\}$.

Denote by Z the set of marked vertices of I. Define $H = G[X \cup Z]$. Notice that $|V(H)| \leq 2|X| + 3\binom{|X|}{2} = \mathcal{O}(k^2)$. This completes the description of the basic compression algorithm that returns H. The key property of H is that it keeps all matching cuts of $G' = G - I_1$, including all matching multicuts of G'. Formally, we define $H' = H - I_1$. At this point, we can observe that the matching multicuts of H' and G' are in a one-to-one correspondence. With a few more technical details, we can prove Theorem 13.

▶ **Theorem 13.** Enum Matching Multicut *admits a polynomial-delay enumeration kernel with $\mathcal{O}(k^2)$ vertices when parameterized by the vertex cover number k of the input graph.*

By Theorem 13, we have that matching multicuts can be listed with delay $k^{\mathcal{O}(k^2)} \cdot n^{\mathcal{O}(1)}$. We believe that this running time can be improved and the dependence on the vertex cover number can be made single exponential.

5.2 Distance to Co-cluster

A 3-approximation for this parameter can easily computed in polynomial time: for every induced $\overline{P_3}$, add all three of its vertices to the modulator. As such we assume that, along with (G, ℓ), we are given a set S of size $k \leq 3\mathsf{dcc}(G)$ so that $G \setminus S$ is a co-cluster graph. We break down our analysis in three cases: if $G \setminus S$ has at least three parts, two large parts, or neither of the previous two. For the first two, we essentially have that $G \setminus S$ is indivisible, while for the last one we may simply invoke the algorithm for the vertex cover parameterization.

▶ **Theorem 14.** Enum Matching Multicut *admits a polynomial-delay enumeration kernel with $\mathcal{O}(k^2)$ vertices when parameterized by the distance to co-cluster k of the input graph.*

5.3 Distance to Cluster

In this section, we present a DelayFPT enumeration algorithm for Enum Matching Multicut, parameterized by the vertex-deletion distance to cluster. We base our result on the quadratic kernel for Matching Cut given in [31]. The authors apply several reduction rules until they reach a kernel of size $\mathrm{dc}(G)^{\mathcal{O}(1)}$. We use a subset of these rules as a starting point for our enumeration algorithm, then expand them a more careful analysis and needed technicalities for an enumeration algorithm. Formally, we prove the following theorem.

▶ **Theorem 15.** *There is an algorithm for Enum Matching Multicut on n-vertex graphs with distance to cluster $\mathsf{dc}(G) \leq t$ of delay $2^{\mathcal{O}(t^3 \log t)} + n^{\mathcal{O}(1)}$.*

Our strategy to enumerate all possible matching multicuts can be divided into 5 steps:

1. We apply reduction rules, similar to the kernel given in [31], spending $\mathsf{poly}(|G|)$ time.
2. We enumerate all possible matching multicuts of a smaller instance of size $\mathcal{O}(t^3)$. This step takes a total time of $2^{\mathcal{O}(t^3 \log t)}$.
3. Given a matching multicut generated in step 2, we create an instance of Enum Set Packing, where the ground set has size t and the number of sets is potentially 2^t. All solutions are enumerated in total time $2^{\mathcal{O}(t^2)}$, and then each solution is extended to form a matching multicut.
4. Given a matching multicut from step 3, we increase the number of partitions by considering clusters of size 2 with only one edge to U. Now, we guarantee that we have at least ℓ partitions. As the number of matching multicuts with at least ℓ parts can be unbounded by ℓ, we worry about the delay of the enumeration and no longer with its total time.
5. We enumerate equivalent solutions for the original instance.

6 Kernelization lower bound for distance to cluster

Since we do have a DelayFPT algorithm for the vertex-deletion distance to cluster parameterization, it is natural to ask whether we can build a PDE kernel of polynomial size. In this section, we show this in the negative by presenting an exponential lower bound for Matching Multicut under this parameterization.

To obtain our result, we first show a kernelization lower bound for Set Packing. In this problem, we are given a ground set X, a family $\mathcal{F} \subseteq 2^X$, and an integer k, and are asked to find $\mathcal{F}' \subseteq \mathcal{F}$ of size at least k such that for any $A, B \in \mathcal{F}'$ it holds that $A \cap B = \emptyset$. In particular, we prove Theorem 16.

▶ **Theorem 16.** *Set Packing has no polynomial kernel when parameterized by $|X|$ unless* $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

Our proof is based on an OR-cross-composition [5] from Set Packing onto itself under the desired parameterization. To this end, we denote our input collection of Set Packing instances by $\{(Y_1, \mathcal{E}_1, r_1), \dots, (Y_t, \mathcal{E}_t, r_t)\}$. Moreover, we can assume that $Y_i = \{y_1, \dots, y_n\}$ and $r_i = r$ for all $i \in [t]$ and, w.l.o.g, that $t = 2^\tau$ for some $\tau > 0$; the latter can be easily achieved by copying any one instance $2^\tau - t$ times and adding it to the input collection, which at most doubles this set if τ is the minimum integer such that $2^\tau \geq t$.

Construction. We construct our $(X, \mathcal{F}, k)$ Set Packing instance as follows. Our set X is partitioned into the set of input elements Y, index elements $S = \{s_0, s_1, \dots, s_r\}$, and a set of bits $\{b_{i,j}, \overline{b}_{i,j} \mid i \in [\tau], j \in [r]\}$. We define $\mathsf{bits}_j(a)$ to be the set where $b_{i,j} \in \mathsf{bits}_j(a)$ if and only if the i-th bit in the binary representation of a is 1, otherwise we have that $\overline{b}_{i,j} \in \mathsf{bits}_j(a)$.

The family $\mathcal{F}$ is partitioned in selector sets, identified as $\mathcal{T} = \{T_1, \ldots, T_t\}$, and packing sets $\mathcal{P}$. Each T_a is defined as $T_a = \{s_0\} \cup \bigcup_{j \in r} \mathsf{bits}_j(\overline{a})$, where $\overline{a}$ is the (positive) bitwise complement of a, i.e. $a + \overline{a} = 2^\tau - 1$. As to our packing sets, for each input instance $(Y, \mathcal{E}_a, r)$, each $C_i \in \mathcal{E}_a$, and each $j \in [r]$, we add to $\mathcal{P}$ the set $C_{a,i,j} = C_i \cup \mathsf{bits}_j(a) \cup \{s_j\}$. Finally, we set $k = r + 1$. Intuitively, packing $T_a \in \mathcal{T}$ corresponds to solving instance $(Y, \mathcal{E}_a, r)$ and, since every T_a has s_0, only one of them can be picked. The way that our bits sets were distributed, picking T_a automatically excludes all elements in $\mathcal{P}$ corresponding to sets present in another instance $(Y, \mathcal{E}_c, r)$. Finally, index elements S are used to ensure that at least one instance set is packed. The next observation follows immediately from the construction of our instance.

▶ **Observation 17.** *Instance $(X, \mathcal{F}, k)$ is such that $|X| \leq |Y| + (r + 1)(1 + \log t)$ and $|C| \leq 1 + r \log t$ for all $C \in \mathcal{F}$.*

The proof of our main result, Theorem 18, follows from a simple polynomial parameter transformation from Set Packing parameterized by the size of the ground set.

▶ **Theorem 18.** *When parameterized by the vertex-deletion distance to cluster, size of the maximum clique, and the number of parts of the cut, Matching Multicut does not admit a polynomial kernel unless* NP ⊆ coNP/poly*.*

7 Final Remarks

In this paper, we introduced and studied the Matching Multicut problem, a generalization of the well known Matching Cut problem, where we want to partition a graph G into at least ℓ parts so that no vertex has more than one neighbor outside of its own part. Specifically, we proved that the problem is NP-hard on subcubic graphs, but admits a quasi-linear kernel when parameterized by ℓ on this graph class. We also showed an $\ell^{\frac{n}{2}} n^{\mathcal{O}(1)}$ exact exponential algorithm based on branching for general graphs. In terms of parameterized complexity, aside from our aforementioned kernel, we give a $2^{\mathcal{O}(t \log t)} n^{\mathcal{O}(1)}$ time algorithm for graphs of treewidth at most t. Then, we move on to enumeration aspects, presenting polynomial-delay enumeration kernels for the vertex cover and distance to co-cluster parameterizations, the latter of which was an open problem for Enum Matching Cut. Finally, we give a DelayFPT algorithm for the distance to cluster parameterization, and show that no polynomial-sized PDE kernel exists unless NP ⊆ coNP/poly. This last result is obtained by showing that Set Packing has no polynomial kernel parameterized by the cardinality of the ground set.

For future work, we are interested in further exploring all aspects of this problem, such as graph classes and other structural parameterizations. As with Matching Cut, it seems interesting to study optimization and perfect variations of this problem, which may yield significant differences in complexity to Matching Multicut. While Maximum Matching Multicut is NP-hard as Perfect Matching Cut is NP-hard on 3-connected cubic planar bipartite graphs [7], the proof does not help in terms of W[1]-hardness. We believe that it in fact is W[1]-hard parameterized by ℓ + number of edges in the cut even on cubic graphs.

Our other questions of interest are mostly in the enumeration realm. In particular, we have no idea if it is possible to enumerate matching cuts on (sub)cubic graphs, and we consider it one of the main open problems in the matching cut literature. Finally, we are interested in understanding how to rule out the existence of TotalFPT and DelayFPT algorithms for a given problem and, ultimately, how to differentiate between problems that admit FPE and PDE kernels of polynomial size and those that do not.

References

1 Júlio Araújo, Nathann Cohen, Frédéric Giroire, and Frédéric Havet. Good edge-labelling of graphs. *Discrete Applied Mathematics*, 160(18):2502–2513, 2012. doi:10.1016/J.DAM.2011.07.021.

2 Rafael T Araújo, Rudini M Sampaio, Vinícius F dos Santos, and Jayme L Szwarcfiter. The convexity of induced paths of order three and applications: complexity aspects. *Discrete Applied Mathematics*, 237:33–42, 2018. doi:10.1016/J.DAM.2017.11.007.

3 N. R. Aravind, Subrahmanyam Kalyanasundaram, and Anjeneya Swami Kare. On structural parameterizations of the matching cut problem. In *Proc. of the 11th International Conference on Combinatorial Optimization and Applications (COCOA)*, volume 10628 of *LNCS*, pages 475–482, 2017. doi:10.1007/978-3-319-71147-8_34.

4 N. R. Aravind and Roopam Saxena. An FPT algorithm for matching cut and d-cut. In Paola Flocchini and Lucia Moura, editors, *Combinatorial Algorithms*, pages 531–543, Cham, 2021. Springer International Publishing. doi:10.1007/978-3-030-79987-8_37.

5 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Cross-composition: A new technique for kernelization lower bounds. In *Proc. of the 28th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 9 of *LIPIcs*, pages 165–176, 2011. doi:10.4230/LIPICS.STACS.2011.165.

6 J.A. Bondy and U.S.R Murty. *Graph Theory*. Springer Publishing Company, Incorporated, 1st edition, 2008.

7 Édouard Bonnet, Dibyayan Chakraborty, and Julien Duron. Cutting barnette graphs perfectly is hard. In Daniël Paulusma and Bernard Ries, editors, *Graph-Theoretic Concepts in Computer Science*, pages 116–129, Cham, 2023. Springer Nature Switzerland. doi:10.1007/978-3-031-43380-1_9.

8 Paul Bonsma. The complexity of the matching-cut problem for planar graphs and other graph classes. *Journal of Graph Theory*, 62(2):109–126, 2009. doi:10.1002/JGT.20390.

9 Andreas Brandstädt and Heinz-Jürgen Voss. Short disjoint cycles in graphs with degree constraints. *Discrete Applied Mathematics*, 64(3):197–205, 1996. doi:10.1016/0166-218X(94)00133-X.

10 Carmen C. Centeno, Simone Dantas, Mitre Costa Dourado, Dieter Rautenbach, and Jayme Luiz Szwarcfiter. Convex partitions of graphs induced by paths of order three. *Discrete Mathematics & Theoretical Computer Science*, 12(Graph and Algorithms), 2010. doi:10.46298/DMTCS.502.

11 Carmen C. Centeno, Mitre C. Dourado, Lucia Draque Penso, Dieter Rautenbach, and Jayme L. Szwarcfiter. Irreversible conversion of graphs. *Theoretical Computer Science*, 412(29):3693–3700, 2011. doi:10.1016/J.TCS.2011.03.029.

12 Chi-Yeh Chen, Sun-Yuan Hsieh, Hoàng-Oanh Le, Van Bang Le, and Sheng-Lung Peng. Matching cut in graphs with large minimum degree. *Algorithmica*, 83:1238–1255, 2021. doi:10.1007/S00453-020-00782-8.

13 Vasek Chvátal. Recognizing decomposable graphs. *Journal of Graph Theory*, 8(1):51–53, 1984. doi:10.1002/JGT.3190080106.

14 Nadia Creignou, Arne Meier, Julian-Steffen Müller, Johannes Schmidt, and Heribert Vollmer. Paradigms for parameterized enumeration. *Theory of Computing Systems*, 60(4):737–758, May 2017. doi:10.1007/s00224-016-9702-4.

15 Marek Cygan, Fedor V Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*, volume 3. Springer, 2015. doi:10.1007/978-3-319-21275-3.

16 Marek Cygan, Paweł Komosa, Daniel Lokshtanov, Marcin Pilipczuk, Michał Pilipczuk, Saket Saurabh, and Magnus Wahlström. Randomized contractions meet lean decompositions. *ACM Trans. Algorithms*, 17(1), December 2021. doi:10.1145/3426738.

17 Mitre C Dourado, Fábio Protti, and Jayme L Szwarcfiter. Complexity results related to monophonic convexity. *Discrete Applied Mathematics*, 158(12):1268–1274, 2010. doi:10.1016/J.DAM.2009.11.016.

18 Rodney G Downey and Michael R Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

19 P. Erdös and L. Pósa. On independent circuits contained in a graph. *Canadian Journal of Mathematics*, 17:347–352, 1965. doi:10.4153/CJM-1965-035-8.

20 Arthur M Farley and Andrzej Proskurowski. Networks immune to isolated line failures. *Networks*, 12(4):393–403, 1982. doi:10.1002/NET.3230120404.

21 Carl Feghali. A note on matching-cut in P_t-free graphs. *Information Processing Letters*, 179:106294, 2023. doi:10.1016/J.IPL.2022.106294.

22 Fedor V. Fomin and Dieter Kratsch. *Exact Exponential Algorithms.* Springer-Verlag, Berlin, Heidelberg, 1st edition, 2010.

23 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: Theory of Parameterized Preprocessing.* Cambridge University Press, 2019.

24 M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified np-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

25 Petr A. Golovach, Christian Komusiewicz, Dieter Kratsch, et al. Refined notions of parameterized enumeration kernels with applications to matching cut enumeration. *Journal of Computer and System Sciences*, 123:76–102, 2022. doi:10.1016/J.JCSS.2021.07.005.

26 Guilherme C. M. Gomes and Ignasi Sau. Finding cuts of bounded degree: Complexity, FPT and exact algorithms, and kernelization. *Algorithmica*, 83(6):1677–1706, June 2021. doi:10.1007/s00453-021-00798-8.

27 Lucía M. González, Luciano N. Grippo, Martín D. Safe, and Vinicius F. dos Santos. Covering graphs with convex sets and partitioning graphs into convex sets. *Information Processing Letters*, 158:105944, 2020. doi:10.1016/J.IPL.2020.105944.

28 Ron L. Graham. On primitive graphs and optimal vertex assignments. *Annals of the New York academy of sciences*, 175(1):170–186, 1970.

29 Frank Harary and Juhani Nieminen. Convexity in graphs. *Journal of Differential Geometry*, 16(2):185–190, 1981.

30 Pinar Heggernes and Jan Arne Telle. Partitioning graphs into generalized dominating sets. *Nord. J. Comput.*, 5(2):128–142, 1998.

31 Christian Komusiewicz, Dieter Kratsch, and Van Bang Le. Matching cut: Kernelization, single-exponential time FPT, and exact exponential algorithms. *Discrete Applied Mathematics*, 283:44–58, 2020. doi:10.1016/j.dam.2019.12.010.

32 Van Bang Le, Felicia Lucke, Daniël Paulusma, and Bernard Ries. Maximizing matching cuts. *arXiv preprint arXiv:2312.12960*, 2023. doi:10.48550/arXiv.2312.12960.

33 László Lovász and Michael D. Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.

34 Cláudio L. Lucchesi and U.S.R. Murty. *Perfect Matchings: A Theory of Matching Covered Graphs.* Algorithms and Computation in Mathematics. Springer Nature Switzerland, Imprint: Springer, 2024.

35 Felicia Lucke, Ali Momeni, Daniël Paulusma, and Siani Smith. Finding d-cuts in graphs of bounded diameter, graphs of bounded radius and h-free graphs. *arXiv preprint arXiv:2404.11389*, 2024. doi:10.48550/arXiv.2404.11389.

36 Felicia Lucke, Daniël Paulusma, and Bernard Ries. On the complexity of matching cut for graphs of bounded radius and H-free graphs. *Theoretical Computer Science*, 936:33–42, 2022. doi:10.1016/J.TCS.2022.09.014.

37 Felicia Lucke, Daniël Paulusma, and Bernard Ries. Dichotomies for maximum matching cut: h-freeness, bounded diameter, bounded radius. *arXiv preprint arXiv:2304.01099*, 2023. doi:10.48550/arXiv.2304.01099.

38 Felicia Lucke, Daniël Paulusma, and Bernard Ries. Finding matching cuts in H-free graphs. *Algorithmica*, 85(10):3290–3322, 2023. doi:10.1007/S00453-023-01137-9.

39 Dáaniel Marx, Barry O'sullivan, and Igor Razgon. Finding small separators in linear time via treewidth reduction. *ACM Trans. Algorithms*, 9(4), October 2013. doi:10.1145/2500119.

40 Augustine M Moshi. Matching cutsets in graphs. *Journal of Graph Theory*, 13(5):527–536, 1989. doi:10.1002/JGT.3190130502.

41 Maurizio Patrignani and Maurizio Pizzonia. The complexity of the matching-cut problem. In *Graph-Theoretic Concepts in Computer Science: 27th InternationalWorkshop, WG 2001 Boltenhagen, Germany, June 14–16, 2001 Proceedings 27*, pages 284–295. Springer, 2001. doi:10.1007/3-540-45477-2_26.

42 Bert Randerath et al. On stable cutsets in line graphs. *Theoretical Computer Science*, 301(1-3):463–475, 2003. doi:10.1016/S0304-3975(03)00048-3.

43 Miklós Simonovits. A new proof and generalizations of a theorem of Erdős and Pósa on graphs without $k+1$ independent circuits. *Acta Mathematica Hungarica*, 18(1-2):191–206, 1967.