Kernelization of Packing Problems

Holger Dell[*] Dániel Marx[†]

Abstract

Kernelization algorithms are polynomial-time reductions from a problem to itself that guarantee their output to have a size not exceeding some bound. For example, d-Set Matching for integers $d \geq 3$ is the problem of finding a matching of size at least k in a given d-uniform hypergraph and has kernels with $O(k^d)$ edges. Recently, Bodlaender et al. [ICALP 2008], Fortnow and Santhanam [STOC 2008], Dell and Van Melkebeek [STOC 2010] developed a framework for proving lower bounds on the kernel size for certain problems, under the complexity-theoretic hypothesis that coNP is not contained in NP/poly. Under the same hypothesis, we show lower bounds for the kernelization of d-Set Matching and other packing problems.

Our bounds are tight for d-Set Matching: It does not have kernels with $O(k^{d-\epsilon})$ edges for any $\epsilon > 0$ unless the hypothesis fails. By reduction, this transfers to a bound of $O(k^{d-1-\epsilon})$ for the problem of finding k vertex-disjoint cliques of size d in standard graphs. It is natural to ask for tight bounds on the kernel sizes of such graph packing problems. We make first progress in that direction by showing non-trivial kernels with $O(k^{2.5})$ edges for the problem of finding k vertex-disjoint paths of three edges each. This does not quite match the best lower bound of $O(k^{2-\epsilon})$ that we can prove.

Most of our lower bound proofs follow a general scheme that we discover: To exclude kernels of size $O(k^{d-\epsilon})$ for a problem in d-uniform hypergraphs, one should reduce from a carefully chosen d-partite problem that is still NP-hard. As an illustration, we apply this scheme to the vertex cover problem, which allows us to replace the number-theoretical construction by Dell and Van Melkebeek [STOC 2010] with shorter elementary arguments.

1 Introduction

Algorithms based on kernelization play a central role in fixed-parameter tractability and perhaps this kind of parameterized algorithms has the most relevance to practical computing. Recall that a problem is *fixed-parameter tractable* parameterized by some parameter k of the instance if it can be solved in time $f(k) \cdot n^{O(1)}$ for some computable function f depending only on the parameter k (see [DF99, FG06, Nie06]). A *kernelization algorithm* for a problem P is a polynomial-time algorithm that, given an instance x of the problem P with parameter k, creates an equivalent instance x' of P such that the size of x' is bounded from above by a function $f(k)$. For example, the classical result of Nemhauser and Trotter [NT74] can be interpreted as a kernelization algorithm that, given an instance of Vertex Cover, produces an equivalent instance on at most $2k$ vertices, which implies that it has at most $\binom{2k}{2}$ edges. A kernelization algorithm can be thought of as preprocessing that creates an equivalent instance whose size has a mathematically provable upper bound that depends only on the parameter of the original instance and not on the size of the original instance. Practical computing often consists of a heuristic preprocessing phase to simplify the instance followed by an exhaustive search for solutions (by whatever method available). Clearly, it is desirable that the preprocessing shrinks the size of the instance as much as possible. Kernelization is a framework in which the efficiency of the preprocessing can be studied in a rigorous way.

One can find several examples in the parameterized complexity literature for problems that admit a kernel with relatively small sizes, i.e., for problems where $f(k)$ is polynomial in k. There are efficient techniques for obtaining such results for particular problems (e.g., [Tho10, Guo09, CFJ04, LMS11, FFL$^+$09]). Some of these techniques go back to the early days of parameterized complexity and have been refined for several years. More recently, general abstract techniques were developed that give us kernelization results for several problems at once [FLST10, BFL$^+$09].

Bodlaender et al. [BDFH09] recently developed a framework for showing that certain parameterized problems are unlikely to have kernels of polynomial size, and Fortnow and Santhanam [FS08] proved the connection with the complexity-theoretic hypothesis coNP $\not\subseteq$ NP/poly. In particular, for several basic problems, such as finding a cycle of length k, a kernelization with polynomial size would imply that coNP $\subseteq$ NP/poly. The framework of Bodlaender et al. [BDFH09] has lead to a long series of hardness results showing that several concrete problems with various

[*]University of Wisconsin–Madison. Research partially supported by NSF grant 1017597 and by the Alexander von Humboldt Foundation.

[†]Institut für Informatik, Humboldt-Universität zu Berlin, Germany, and Computer and Automation Research Institute, Hungarian Academy of Sciences (MTA SZTAKI), Budapest, Hungary.

parameterizations are unlikely to have kernels of polynomial size [CFM11, BTY09, DLS09, FFL$^+$09, KW10, KW09, KMW10, BJK11b, BJK11a, MRS11].

More recently, Dell and Van Melkebeek [DvM10] refined the complexity results of [FS08, BDFH09] to prove conditional lower bounds also for problems that do admit polynomial kernels. For example, they show that Vertex Cover does not have kernels of size $O(k^{2-\epsilon})$ unless the hypothesis coNP $\not\subseteq$ NP/poly, which is the same as above, fails. Similar lower bounds are given for several other graph covering problems where the goal is to delete the minimum number of vertices in such a way that the remaining graph satisfies some prescribed property. Many of the lower bounds are tight as they match the upper bounds of the best known kernelization algorithms up to an arbitrarily small ϵ term in the exponent.

In the present paper, we also obtain kernel lower bounds for problems that have polynomial kernels, but the family of problems that we investigate is very different: packing problems. Covering and packing problems are dual to each other, but there are significant differences in the way they behave with respect to fixed-parameter tractability. For example, techniques such as bounded search trees or iterative compression are mostly specific to covering problems, while techniques such as color coding are mostly specific to packing problems. Feedback Vertex Set is the problem of covering all cycles and has kernels with $O(k^2)$ edges [Tho10, DvM10], while its packing version is the problem of finding k vertex-disjoint cycles and is unlikely to have polynomial kernels [BTY09]. Therefore, the techniques for understanding the kernelization complexity of covering and packing problems are expected to differ very much. Indeed the proofs in [DvM10] for the problem of covering sets of size d cannot be straightforwardly adapted to the analogous problem of packing sets of size d.

Our contributions are twofold. First, we obtain lower bounds on the kernel size for packing sets and packing disjoint copies of a prescribed subgraph H. An example of the latter is the problem of finding k vertex-disjoint d-cliques in a given graph. For packing sets, our lower bound is tight, while determining the best possible kernel size for graph packing problems with every fixed H remains an interesting open question. Fully resolving this question would most certainly involve significantly new techniques both on the complexity and the algorithmic side. To indicate what kind of difficulties we need to overcome for the resolution of this question, we show kernels with $O(k^{2.5})$ edges for the problem of packing k vertex-disjoint paths on four vertices.

Secondly, the techniques used in our lower bounds are perhaps as important as the concrete results themselves. We present a simple and clean way of obtaining lower bounds of the form $O(k^{d-\epsilon})$. Roughly speaking, the idea is to reduce from an appropriate d-partite problem by observing that if we increase the size of the universe by a factor of $t^{1/d}$, then we can conveniently pack together t instances. A similar effect was achieved in [DvM10], but it used a combinatorial tool called the *Packing Lemma*, whose proof uses nontrivial number-theoretical arguments. As a demonstration, we show that our scheme allows us to obtain the main kernelization results of [DvM10] with very simple elementary techniques. Furthermore, this scheme proves to be very useful for packing problems, even though in one of our lower bounds it was easier to invoke the Packing Lemma. It seems that both techniques will be needed for a complete understanding of graph packing problems.

1.1 Results.

The matching problem in d-uniform hypergraphs, d-Set Matching, is to decide whether a given hypergraph has a matching of size k, i.e., a set of k pairwise disjoint hyperedges. Correspondingly, the Perfect d-Set Matching problem is to find a *perfect* matching, i.e., a matching with $k = n/d$ where n is the number of vertices. Fellows et al. [FKN$^+$08] show that d-Set Matching has kernels with $O(k^d)$ hyperedges.

Theorem 1.1 ([FKN$^+$08]). *The problem d-Set Matching has kernels with $O(k^d)$ hyperedges.*

In Appendix A, we sketch a straightforward but instructive proof of this fact using the sunflower lemma of Erdős and Rado [ER60]. Our main result is that the kernel size above is asymptotically optimal under the hypothesis coNP $\not\subseteq$ NP/poly.

Theorem 1.2. *Let $d \geq 3$ be an integer and ϵ a positive real. Then Perfect d-Set Matching does not have kernels of size $O(k^{d-\epsilon})$ unless* coNP $\subseteq$ NP/poly.

Since Perfect d-Set Matching is a special case of d-Set Matching, the lower bound applies to that problem as well and it shows that the upper bound in Theorem 1.1 is asymptotically tight.

A particularly well-studied special case of set matching is when the sets are certain fixed subgraphs (e.g., triangles, cliques, stars, etc.) of a given graph. We use the terminology of Yuster [Yus07], who surveys graph theoretical properties of such graph packing problems. Formally, an *H-matching* of size k in a graph G is a collection of k vertex-disjoint subgraphs of G that are isomorphic to H. The problem H-Matching is to find an H-matching of a given size in a given graph. Both problems are NP-complete whenever H contains a connected component with more than two vertices [KH78] and is in P otherwise.

The kernelization properties of graph packing problems received a lot of attention in the literature (e.g., [Mos09, FHR+04, PS06, FR09, WNFC10, MPS04]). H-MATCHING can be expressed as a d-SET MATCHING instance with $O(k^d)$ edges (where $d := |V(H)|$) and therefore Theorem 1.1 implies a kernel of size $O(k^d)$. In the particularly interesting special case when H is a clique K_d, we use a simple reduction to transfer the above theorem to obtain a lower bound for K_d-MATCHING.

Theorem 1.3. *Let $d \geq 4$ be an integer and ϵ a positive real. Then K_d-MATCHING does not have kernels of size $O(k^{d-1-\epsilon})$ unless* coNP $\subseteq$ NP/poly.

An upper bound of size $O(k^d)$ follows for K_d-MATCHING from Theorem 1.1. This does not quite match our conditional lower bounds of $O(k^{d-1-\epsilon})$, and it is an interesting open problem to make the bounds tight.

The H-FACTOR problem is the restriction of H-MATCHING to the case $k = n/d$, i.e., the goal is to find an H-matching that involves all vertices. Unlike the case of matching d-sets, where we had the same bounds for PERFECT d-SET MATCHING and d-SET MATCHING, we cannot expect that the same bounds hold always for H-MATCHING and H-FACTOR. The reason is that for H-FACTOR there is a trivial $O(k^2)$ upper bound on the kernel size for *every* graph H: an n-vertex instance has size $O(n^2)$ and we have $k = \Theta(n)$ by the definition of H-FACTOR. We show that this bound is tight for every NP-hard H-FACTOR problem. Thus, we cannot reduce H-FACTOR to sparse instances. The proof of this result is based on the Packing Lemma of [DvM10].

Theorem 1.4. *Let H be a connected graph with $d \geq 3$ vertices and ϵ a positive real. Then H-FACTOR does not have kernels of size $O(k^{2-\epsilon})$ unless* coNP $\subseteq$ NP/poly.

Obviously, Theorem 1.4 gives a lower bound for the more general H-MATCHING problem. In particular, it proves the missing $d = 3$ case in Theorem 1.3.

Obtaining tight bounds for H-MATCHING seems to be a challenging problem in general. As Theorem 1.3 shows in the case of cliques, the lower bound of $O(k^{2-\epsilon})$ implied by Theorem 1.4 is not always tight. We demonstrate that the upper bound of $O(k^{|V(H)|})$ is not always tight either. A simple argument shows that if H is a star of arbitrary size, then a kernel of size $O(k^2)$ is possible, which is tight by Theorem 1.4. Furthermore, if H is a path on 3 edges, then a surprisingly nontrivial extremal argument gives us the following.

Theorem 1.5. *P_3-MATCHING has kernels with $O(k^{2.5})$ edges.*

The examples of cliques, stars, and paths show that the exact bound on the kernel size of H-MATCHING for a particular H could be very far from the weak $O(k^{|V(H)|})$ upper bound or the weak $O(k^{2-\epsilon})$ lower bound (Theorem 1.4). Full understanding of this question seems to be a very challenging, yet very natural problem. Our proof of Theorem 1.5 might indicate what kind of combinatorial problems we have to understand for a full solution.

After obtaining our results, we learnt that Hermelin and Wu [HW11] also achieved kernel lower bounds for packing problems using the paradigm of Lemma 2.1. In particular, their bound for d-SET MATCHING is $O(k^{d-3-\epsilon})$ and it is $O(k^{d-4-\epsilon})$ for K_d-MATCHING.

2 Techniques

The OR *of a language* L is the language OR(L) that consists of all tuples $(x_1, \ldots, x_t)$ for which there is an $i \in [t]$ with $x_i \in L$. Instances $\overline{x} = (x_1, \ldots, x_t)$ for OR(L) have two natural parameters: the length t of the tuple and the maximum bitlength $s = \max_i |x_i|$ of the individual instances for L. The following lemma captures the method that was used in [DvM10] to prove conditional kernel lower bounds.

Lemma 2.1. *Let Π be a problem parameterized by k and let L be an* NP*-hard problem. Assume that there is a polynomial-time mapping reduction f from* OR(L) *to Π and a number $d > 0$ with the following property: given an instance $\overline{x} = (x_1, \ldots, x_t)$ for* OR(L) *in which each x_i has size at most s, the reduction produces an instance $f(\overline{x})$ for Π whose parameter k is at most $t^{1/d+o(1)} \cdot$* poly(s).

Then L does not have kernels of size $O(k^{d-\epsilon})$ for any $\epsilon > 0$ unless coNP $\subseteq$ NP/poly.

Bodlaender et al. [BDFH09] formulated this method without the dependency on t. This suffices to prove polynomial kernel lower bounds since d can be chosen as an arbitrarily large constant. It was observed in [DvM10] that the proofs in [BDFH09, FS08] can be easily adapted to obtain the formulation above, and that it can be generalized to an oracle communication setting.

We now informally explain a simple scheme for proving kernel lower bounds of the form $O(k^{d-\epsilon})$ for a parameterized problem Π. Lemma 2.1 requires us to devise a reduction from OR(L) (for some NP-hard language L) to Π whose output instances have parameter k at most $t^{1/d} \cdot$ poly(s). We carefully select a problem L whose definition is d-partite in a certain sense, and we design the reduction from OR(L) to Π using the general scheme described. Most problem parameters can be bounded from above by the number of vertices; therefore, what we need to ensure is that the number of vertices increases roughly by at most a factor of $t^{1/d}$.

For simplicity of notation, we informally describe the case $d = 2$ first. We assume that L is a bipartite problem, meaning that each instance is defined on two sets U and W, and "nothing interesting is happening inside U or inside W." We construct the instance of Π by taking $\sqrt{t}$ copies of U and $\sqrt{t}$ copies of W. For every copy of U and every copy of W, we embed one of the t instances appearing in the OR(L) instance. This way, we can embed $\sqrt{t} \cdot \sqrt{t} = t$ instances, as required. The fact that L is a bipartite problem helps ensuring that two instances of L sharing the same copy of U or the same copy of W do not interfere. A crucial part of the reduction is to ensure that every solution of the constructed instance can use at most one copy of U and at most one copy of W. If we can maintain this property (using additional arguments or introducing gadgets), then it is usually easy to show that the constructed instance has a solution if and only if at least one of the $\sqrt{t} \cdot \sqrt{t}$ instances appearing in its construction has a solution.

For $d > 2$, the scheme is similar. We start with a d-partite problem L and make $t^{1/d}$ copies of each partition class. Then there are $(t^{(1/d)})^d = t$ different ways of selecting one copy from each class, and therefore we can compose together t instances following the same scheme.

As a specific example, let us consider Π = Vertex Cover in graphs, where we have $d = 2$. We demonstrate that the lower bound for this problem can be proved elegantly if we make the not completely obvious choice of selecting L to be Multicolored Biclique:

Input: A bipartite graph B on the vertex set $U \dot\cup W$, an integer k, and partitions $U = (U_1, \ldots, U_k)$ and $W = (W_1, \ldots, W_k)$.

Decide: Does B contain a biclique $K_{k,k}$ that has one vertex from each U_a and W_a $(1 \le a \le k)$?

This is a problem on bipartite graphs and NP-complete as we prove in Appendix B.

Theorem 2.1 ([DvM10]). Vertex Cover *does not have kernels of size $O(k^{2-\epsilon})$ unless* coNP $\subseteq$ NP/poly.

Proof. We apply Lemma 2.1 where we set L = Multicolored Biclique. Given an instance $(B_1, \ldots, B_t)$ for OR(L), we can assume that every instance B_i has the same number k of groups in the partitions and every group in every instance B_i has the same size n: by simple padding, we can achieve this property in a way that increases the size of the OR(L) instance by at most a polynomial factor. Furthermore, we can assume that $\sqrt{t}$ is an integer. In the following, we refer to the t instances of Multicolored Biclique in the OR(L) instance as $B_{(i,j)}$ for $1 \le i, j \le \sqrt{t}$; let $U_{(i,j)}$ and $W_{(i,j)}$ be the two bipartite classes of $B_{(i,j)}$.

First, we modify each instance $B_{(i,j)}$ in such a way that $U_{(i,j)}$ and $W_{(i,j)}$ become complete k-partite graphs: if two vertices $U_{(i,j)}$ or two vertices in $W_{(i,j)}$ are in different groups, then we make them adjacent. It is clear that there is a $2k$-clique in the new graph $B'_{(i,j)}$ if and only if there is a correctly partitioned $K_{k,k}$ in $B_{(i,j)}$.

We construct a graph G by introducing $2\sqrt{t}$ sets $U^1, \ldots, U^{\sqrt{t}}, W^1, \ldots, W^{\sqrt{t}}$ of kn vertices each. For every $1 \le i \le j \le \sqrt{t}$, we copy the graph $B'_{(i,j)}$ to the vertex set $U^i \cup W^j$ by mapping $U_{(i,j)}$ to U^i and $W_{(i,j)}$ to W^j. Note that $U_{(i,j)}$ and $W_{(i,j)}$ induces the same complete k-partite graph in $B'_{(i,j)}$ for every i and j, thus this copying can be done in such a way that $G[U^i]$ receives the same set of edges when copying $B'_{(i,j)}$ for any j (and similarly for $G[W^j]$). Therefore, $G[U^i \cup W^j]$ is isomorphic to $B'_{(i,j)}$ for every $1 \le i, j \le \sqrt{t}$.

We claim that G has a $2k$-clique if and only if at least one $B'_{(i,j)}$ has a $2k$-clique (and therefore at least one $B_{(i,j)}$ has a correctly partitioned $K_{k,k}$). The reverse direction is clear, as $B'_{(i,j)}$ is a subgraph of G by construction. For the forward direction, observe that G has no edge between U^i and $U^{i'}$, and between W^j and $W^{j'}$ for any $i \ne i'$ or $j \ne j'$. Therefore, the $2k$-clique of G is fully contained in $G[U^i \cup W^j]$ for some $1 \le i, j \le \sqrt{t}$. As $G[U^i \cup W^j]$ is isomorphic to $B'_{(i,j)}$, this means that $B'_{(i,j)}$ also has a $2k$-clique.

Let $N = 2\sqrt{t} \cdot kn$ be the number of vertices in G. Note that $N = t^{1/2} \cdot \text{poly}(s)$, where s is the maximum bitlength of the t instances in the OR(L) instance. The graph G has a $2k$-clique if and only if its complement $\overline{G}$ has a vertex cover of size $N - 2k$. Thus OR(L) can be reduced to an instance of Vertex Cover with parameter at most $t^{1/2} \cdot \text{poly}(s)$, as required. ∎

In Appendix C, we transfer the above ideas to the vertex cover problem for d-uniform hypergraphs.

3 Kernelization of the Set Matching Problem

The d-Set Matching problem is to find a maximum collection of hyperedges in a d-uniform hypergraph such that any two hyperedges are disjoint. For $d = 2$, this is the maximum matching problem and polynomial-time solvable. The restriction of this problem to d-partite hypergraphs is the d-dimensional matching problem and NP-hard [Kar72] for $d \ge 3$.

We use Lemma 2.1 to prove that the kernel size in Theorem 1.1 is asymptotically optimal under the hypothesis coNP $\not\subseteq$ NP/poly. For the reduction, we use gadgets with few vertices that coordinate the availability of groups of vertices. For example, we may have two sets U_1, U_2 of vertices and our gadget makes sure that in every perfect packing of the graph one set is fully covered by the gadget while the other group has to be covered by hyperedges of the graph external to the gadget. Ultimately, this enables us to choose between different instances in the OR-problem. The precise formulation of the gadget is as follows.

Lemma 3.1. *Let $d \geq 3$, $m \geq 1$, and $s \geq 1$ be integers. In time polynomial in d, m, s, we can compute a d-uniform hypergraph S with $O(dsm)$ vertices and pairwise disjoint sets $U_1(S), \ldots, U_m(S) \subset V(S)$ of size s each, such that the following conditions hold.*

(i) (Completeness) For each i, $S - U_i$ has a perfect matching.

(ii) (Soundness) If S is a subgraph of some G and the vertices of $S - (U_1 \cup \cdots \cup U_m)$ are only contained in edges of S, then every perfect matching of G contains a perfect matching of $S - U_i$ for some i.

(iii) The underlying graph of S (the graph obtained by replacing the d-hyperedges of S by d-cliques) does not contain a clique of size $d + 1$ and it contains $\bigcup_i U_i$ as an independent set.

In addition to the completeness and the soundness properties that make the gadget work the way we want, we also have a structural property (iii), which we need later when we transfer our results to K_d-MATCHING. We defer the proof of Lemma 3.1 to the end of this section and use it now to prove the following.

Lemma 3.2. *For any integer $d \geq 3$, there is a $\leq^p_m$-reduction from* OR(d-SET MATCHING) *to d-*SET MATCHING *that maps t-tuples of instances of bitlength s each to instances on $t^{1/d} \cdot \mathrm{poly}(s)$ vertices whose underlying graph does not contain a clique of size $d + 1$.*

Proof. Let $G_1, \ldots, G_t$ be instances of d-SET MATCHING, i.e., d-uniform hypergraphs of size s each. Finding perfect matchings in d-partite d-uniform hypergraphs is NP-hard for $d \geq 3$, so we can assume w.l.o.g. that the G_i's are d-partite and each part of the partition contains exactly s/d vertices. The goal is to find out whether some G_i contains a perfect matching. We reduce this question to an instance G on few vertices.

The vertex set of G consists of $d \cdot t^{1/d}$ groups of n/d vertices each, i.e., $V(G) = \bigcup_{a,b} V_{a,b}$ for $a \in [d]$ and $b \in [t^{1/d}]$. Then we can write the input graphs as G_b using an index vector $b = (b_1, \ldots, b_d) \in [t^{1/d}]^d$. For each graph G_b we add edges to G in the following way: We identify the vertex set of G_b with $V_{1,b_1} \dot\cup \ldots \dot\cup V_{d,b_d}$, and we let G contain all the edges of G_b. Since each G_b is d-partite, the same is true for G at this stage of the construction. Now we modify G such that each perfect matching of G only ever uses edges originating from at most one graph G_b. For this it suffices to add a gadget for every $a \in [d]$ that blocks all but exactly one group $V_{a,b}$ in every perfect matching. For each $a \in [d]$, we add a copy S_a of $S(V_{a,1}, \ldots, V_{a,m})$ from Lemma 3.1 to G, where $m = t^{1/d}$. Clearly, $|V(G)| \leq O(st^{1/d})$. Furthermore, the underlying graph of G does not contain a clique of size $d+1$ as the graph restricted to $\bigcup_{a,b} V_{a,b}$ is d-partite and the gadgets do not contain cliques of size $d + 1$ in their underlying graph.

Now we verify the correctness of the reduction. If some G_b has a perfect matching then the completeness property of S_a ensures that $S_a - V_{a,b_a}$ has a perfect matching for all $a \in [d]$. Together with the perfect matching of G_b this gives a perfect matching of G. For the soundness, assume M is a perfect matching of G. Then each S_a is guaranteed to have a b_a such that M contains a perfect matching of $S_a - V_{a,b_a}$. Since V_{a,b_a} is an independent set in S_a, M uses only edges of G_b to cover the V_{a,b_a}. In particular, G_b has a perfect matching. ■

Theorem 1.2, our kernel lower bound for d-SET MATCHING, now follows immediately by combining the above with Lemma 2.1.

3.1 Proof of Lemma 3.1.

We use cycles as building blocks in the gadget constructions. A *loose cycle of length ℓ* in a d-uniform hypergraph is a sequence $C = v_1, e_1, v_2, e_2, \ldots, v_\ell, e_\ell$ with the property that $e_i \cap e_{i+1} = \{v_{i+1}\}$ and $e_i \cap e_j = \emptyset$ if $i \notin \{j-1, j, j+1\}$. The indices are always understood modulo ℓ. The vertices $v_1, \ldots, v_\ell$ are the *connection* vertices, whereas all other vertices are *free* vertices of the cycle. Our first lemma, which allows us to coordinate two sets of vertices.

Lemma 3.3. *Let $d \geq 3$ and $s \geq 1$ be integers. Let $C = v_1, e_1, v_2, e_2, \ldots, v_{2s}, e_{2s}$ be a loose cycle of d-hyperedges as depicted in Figure 1 for $s = 3$. We define $U_1(C) = \bigcup_{i \text{ even}} e_i \setminus \{v_i, v_{i+1}\}$ and $U_2(C) = \bigcup_{i \text{ odd}} e_i \setminus \{v_i, v_{i+1}\}$. Then*

(i) (Completeness) $C - U_1$ and $C - U_2$ have a perfect matching.

(ii) (Soundness) If C is a subgraph of some G and the vertices of $C - (U_1 \cup U_2)$ are only contained in edges of C, then every perfect matching of G contains a perfect matching of $C - U_i$ for some i.

Proof. For the completeness, $\{e_{2i+1}\}$ forms a perfect matching of $C - U_1$ and $\{e_{2i}\}$ forms a perfect matching of $C - U_2$. For the soundness, the only way to cover

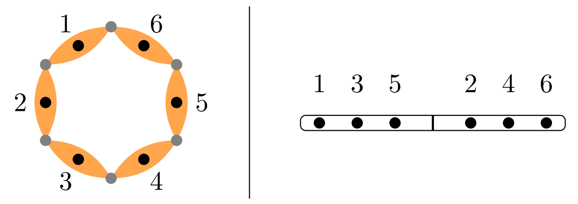

Figure 1: *Left:* An even cycle gadget with $d = 3$, $s = 3$, $U_1 = \{1, 3, 5\}$, and $U_2 = \{2, 4, 6\}$. Black vertices are free vertices, and gray vertices are connection vertices that are not supposed to be adjacent to any other vertex of the outside graph. *Right:* Pictorial abbreviation of the graph on the left. By Lemma 3.3, any perfect matching blocks exactly the vertices in one of the halves using edges of the gadget.

a vertex v_i of C is to pick one of its two incident hyperedges. Since C is an even cycle, the two ways of doing this for all such vertices in a consistent way are as in the completeness step. ■

We use the above gadget with two choices to construct the gadget in Lemma 3.1, which forces perfect matchings to choose properly between m sets of vertices.

Proof (of Lemma 3.1). We construct a *coordination gadget* S as depicted in Figure 2 as follows:

1. We start with s disjoint odd cycles: loose cycles $C_1, \ldots, C_s$ of length $2m + 1$ each. We denote the vertices in these cycles with $c_{i,j}$ and the edges with $C_{i,j}$, i.e.,

$$C_i = c_{i,1}, C_{i,1}, c_{i,2}, C_{i,2}, \ldots, c_{i,2m+1}, C_{i,2m+1}.$$

   Let $C = \bigcup_{i,j} C_{i,j} \setminus \{c_{i,j}, c_{i,j+1}\}$ be the set of all free vertices in these cycles.

2. We define $U_j(S) = \{c_{1,2j}, \ldots, c_{s,2j}\}$ for all $j \in [m]$.

3. We add $2m + 1$ disjoint even cycles: loose cycles $F_1, \ldots, F_{2m+1}$ of length $2s$ as in Lemma 3.3. We denote the vertices in these cycles with $f_{j,i}$ and the edges with $F_{j,i}$, i.e.,

$$F_j = f_{j,1}, F_{j,1}, f_{j,2}, F_{j,2}, \ldots, f_{j,2s}, F_{j,2s}.$$

   We identify $\bigcup_j U_1(F_j)$ and C in such a way that

$$F_{j,2i} \setminus \{f_{j,2i}, f_{j,2i+1}\} = C_{i,j} \setminus \{c_{i,j}, c_{i,j+1}\}. \tag{3.1}$$

   Let $F = \bigcup_{j,i} F_{j,i} \setminus \{f_{j,i}, f_{j,i+1}\}$ be the set of all free vertices in the even cycles.

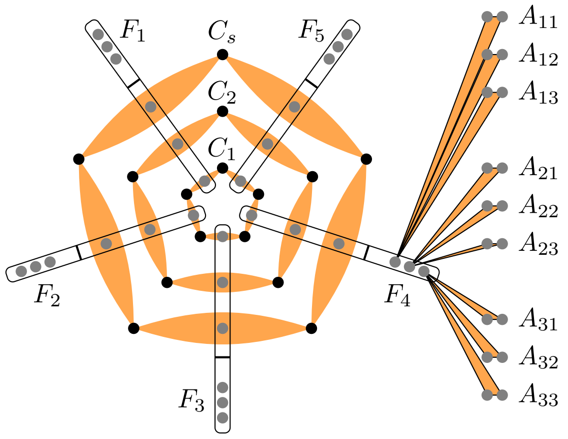

Figure 2: A coordination gadget as in Lemma 3.1 for $d = 3$, $s = 3$ and $m = 2$. All vertices are drawn, and all edges of the odd cycles C_i. The boxes for the F_j represent even cycle gadgets from Figure 1. All edges between the $A_{k,\ell}$ and F_4 are drawn, but all other edges incident to the $A_{k,\ell}$ are omitted. They attach to the other F_j's in the same fashion.

4. For $j \in [2m + 1]$, enumerate the vertices $v_{j,1}, \ldots, v_{j,(d-2)s}$ of $U_2(F_j) = V(F_j) \cap F \setminus C$ arbitrarily. For each $k \in [(d - 2)s]$ and $\ell \in [m + 1]$, add a set $A_{k,\ell}$ of $|A_{k,\ell}| = d - 1$ fresh vertices and add the "saturation" hyperedges $A_{k,\ell} \cup \{v_{j,k}\}$ to S for all choices of $j \in [2m + 1]$.

This finishes the construction of S. First we show (iii). For this we consider the underlying graph and assume for contradiction that T is a clique of size $d+1$. By the way S was constructed, and in particular by (3.1), each hyperedge of S intersects at most one set of free vertices that belongs to some cycle edge, so any two vertices from distinct sets of free vertices must be non-adjacent in the underlying hypergraph. To reach a contradiction, we distinguish two cases.

Case 1: T contains a vertex $v \in A_{k,\ell}$ for some k, ℓ. Since v's only neighbors are the $d - 2$ other vertices of $A_{k,\ell}$ and the vertices $v_{j,k}$, T contains $v_{j,k}$ and $v_{j',k}$ for $j \neq j'$. However, these vertices are not adjacent since they belong to different even cycles.

Case 2: T contains only vertices of the cycles. Then T must contain a connection vertex v of one of the cycles since any free vertex is adjacent to at most $d - 3$ other free vertices. The vertex v is adjacent to exactly $2d - 2$ vertices, and so T contains a free vertex w in the edge before v and w' in the edge after v in the respective cycle. By the above, w and w' are not adjacent.

This shows that the underlying graph does not contain a $(d+1)$-clique. For the second part, we observe that $\bigcup_{i\in[m]} U_i$ is the set of connection vertices at even positions of the odd cycles, so they are pairwise non-adjacent.

For the completeness, we construct a perfect matching of $S - U_{j_0}(S)$ for each $j_0 \in [m]$. We define the set of indices

(3.2)
$$J = \left\{ 2j_0 + 2j \;\middle|\; j = 0, \dots, m \right\} .$$

We use the completeness of the even cycle gadgets and take a perfect matching of F_j that covers $U_1(F_j)$ for all $j \in J$, and one that covers $U_2(F_j)$ for the m other choices $j \in [2m+1] \setminus J$. This is consistent since the even cycles are disjoint. In each odd cycle C_i, we pick the edges $C_{i,j}$ into the matching for $j \in [2m+1]\setminus J$. This is consistent because these edges do not contain a vertex of $U_{j_0}(S)$ or of $U_1(F_j)$ for $j \in J$, and we never take two consecutive edges. Furthermore, we have covered all vertices of $C - U_{j_0}(S)$. Indeed, the only vertices not yet covered are the $U_2(F_j) = \{v_{j,1}, \dots, v_{j,(d-2)s}\}$ for $j \in J$ and the vertices of the $A_{k,\ell}$. For each $k \in [(d-2)s]$ and $j \in J$, we cover the vertex $v_{j,k}$ using a saturation edge with some $A_{k,\ell}$. This is possible and covers all $A_{k,\ell}$ since each k has exactly $|J| = m+1$ disjoint groups of $A_{k,\ell}$. Now all vertices of $S - U_{j_0}$ are covered by a perfect matching.

For the soundness, the claim is that any perfect matching of G has some j_0 such that U_{j_0} is not covered in the matching by edges of S, whereas all other vertices of S are. Let M be a perfect matching of G. The soundness of the even cycle gadgets guarantees that exactly one of $U_1(F_j)$ and $U_2(F_j)$ are covered with edges of F_j. Let J be the set of indices j for which $U_1(F_j)$ and not $U_2(F_j)$ is covered by the edges of F_j. The only way that M can cover the vertices $U_2(F_j)$ for $j \in J$ is by using $|U_2(F_j)| = (d-2)s$ edges with the $A_{k,\ell}$'s. Since there are only $m+1$ such edges available for any given k, we have $|J| = m+1$. The only way that M can cover the free vertices of $C_{i,j}$ for $j \in [2m+1] \setminus J$ is by picking $C_{i,j}$ into M. Since M does not contain consecutive edges of C_i and J contains $m+1$ elements of $[2m+1]$, this means that J must be of the form (3.2) for some j_0. Hence $U_j(S)$ for $j \neq j_0$ is covered in M by edges of the odd cycles and no vertex of U_{j_0} is covered in M by edges of S. ∎

4 Kernel Lower Bounds for Graph Matching Problems

For a graph H, the H-matching problem is to find a maximal number of vertex-disjoint copies of H in a given graph G. This problem is NP-complete whenever H contains a connected component with more than two vertices [KH78] and is in P otherwise.

4.1 Clique Packing. We prove Theorem 1.3, that K_d-Matching for $d \geq 4$ does not have kernels of size $O(k^{d-1-\epsilon})$ unless coNP $\subseteq$ NP/poly. For this, we devise a parameter-preserving reduction from the problem of finding a perfect matching in a $(d-1)$-uniform hypergraph whose underlying graph does not contain a d-clique.

Lemma 4.1. *Let $d \geq 4$ be an integer. There is a $\leq_m^p$-reduction from $(d-1)$-Set Matching in $(d-1)$-uniform hypergraphs whose underlying graph does not contain a clique of size d to K_d-Matching that does not change the parameter k.*

Proof. Let G be a $(d-1)$-uniform hypergraph on n vertices without d-clique in its underlying graph. For each edge e of G, we add a new vertex v_e and transform $e \cup \{v_e\}$ into a d-clique in G'. We claim that G has a matching of size $k := n/(d-1)$ if and only if G' has a K_d-matching of size k. The completeness is clear since any given matching of G can be turned into a K_d-matching of G' by taking the respective d-clique for every $(d-1)$-hyperedge. For the soundness, let G' contain a K_d-matching of size k. Note that any d-clique of G' uses exactly one vertex v_e since the underlying graph of G does not contain any d-cliques and since no two v_e's are adjacent. Thus every d-clique of G' is of the form $e \cup \{v_e\}$, which gives rise to a matching of G of size k. ∎

This combined with Lemma 2.1 and Lemma 3.2 implies Theorem 1.3

4.2 General Graph Matching Problems. We prove Theorem 1.4, that H-factor does not have kernels of size $O(k^{2-\epsilon})$ unless coNP $\subseteq$ NP/poly, whenever H is a connected graph with at least three vertices. In particular, this implies the missing case $d = 3$ of K_d-Matching.

We use the coordination gadget of Lemma 3.1 in a reduction from a suitable OR-problem to H-Matching. To do so, we translate the coordination gadget for Perfect d-Set Matching to H-factor, which we achieve by replacing hyperedges with the following hyperedge-gadgets of [KH78].

Lemma 4.2. *Let H be a connected graph on $d \geq 3$ vertices. There is a graph $e = e(v_1, \dots, v_d)$ that contains $\{v_1, \dots, v_d\}$ as an independent set such that, for all $S \subseteq \{v_1, \dots, v_d\}$, the graph $e - S$ has an H-factor if and only if $|S| = 0$ or $|S| = d$.*

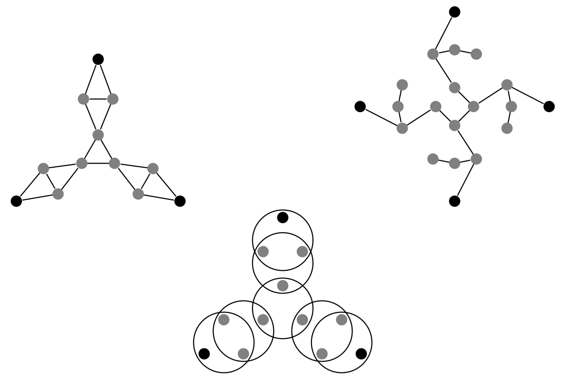

Figure 3: Hyperedge gadgets for different H-matching problems. The outermost, black vertices are the vertices of the simulated hyperedge and the gray vertices are not supposed to be adjacent to any other vertex of the graph. *Left:* Triangle matching. *Middle:* 3-Path matching. *Right:* The general case; each circle represents a copy of H.

Proof. Let v be a vertex of H. We construct e as in Figure 3. We start with one central copy of H. For each vertex $u \in [d] = V(H)$, we create a new copy H_u of H and denote its copy of v by v_u. Finally, we add an edge between $u \in H$ and $w \in (H_u - v_u)$ if $v_u w$ is an edge of H_u.

For the claim, assume that $0 < |S| < d$. Then $|V(e - S)|$ is not an integer multiple of $d = |V(H)|$ and there can be no H-factor in $e - S$. For the other direction, assume that $|S| = 0$. Then the subgraphs H_u for $u \in [d]$ and H are $d+1$ pairwise disjoint copies of H in e and form an H-factor of e. In the case $|S| = d$, we observe that the d subgraphs $(H_u - v_u) \cup \{u\}$ form an H-factor of $e - S = e - \{v_1, \dots, v_d\}$. ∎

For the proof of the H-Packing kernel lower bounds, we need the Packing Lemma.

Lemma 4.3 (Packing Lemma [DvM10]). *For any integers $p \geq d \geq 2$ and $t > 0$ there exists a p-partite d-uniform hypergraph P on $O\big(p \cdot \max(p, t^{1/d+o(1)})\big)$ vertices such that*

(i) the hyperedges of P partition into t cliques $K_1, \dots, K_t$ on p vertices each, and

(ii) P contains no cliques on p vertices other than the K_i's.

Furthermore, for any fixed d, the hypergraph P and the K_i's can be constructed in time polynomial in p and t.

The chromatic number $\chi(H)$ is the minimum number of colors required in a proper vertex-coloring of H. The proof of [KH78] shows that H-Factor is NP-complete even in case we are looking for an H-factor in $\chi(H)$-partite graphs. We are going to make use of that in the following reduction.

Lemma 4.4. *There is a $\leq_m^p$-reduction from* OR(H-Factor) *to* H-Factor *that maps t-tuples of instances of size s each to instances that have at most $\sqrt{t}^{1+o(1)} \cdot \mathrm{poly}(s)$ vertices.*

Proof. Let $p = \chi(H)$ be the chromatic number of H. For an instance $G_1, \dots, G_t$ of OR(H-Factor), we can assume w.l.o.g. that the G_i are p-partite graphs with n vertices in each part. We construct a graph G that has an H-factor if and only if some G_i has an H-factor. For this, we invoke the Packing Lemma, Lemma 4.3, with $d = 2$, and we obtain a p-partite graph P that contains t cliques $K_1, \dots, K_t$ on p vertices each. We identify the vertex set of G_i with $V(K_i) \times [n]$ injectively in such a way that vertices in the same color class have the same first coordinate. We define an intermediate p-partite graph G' on the vertex set $V(P) \times [n]$ as $G' = G_1 \cup \dots \cup G_t$. To obtain G from G', we add p coordination gadgets of Lemma 3.1 with $m = \sqrt{t}^{1+o(1)}$ and $d = p$. For each color class $C \subset V(G')$, we add a coordination gadget where the $U_i \subset C$ are those vertices that project to the same vertex in P. Finally, we replace each p-hyperedge by the gadget in Lemma 4.2, which finishes the construction of G.

For the completeness of the reduction, assume G_i has an H-factor M. To construct an H-factor of G, we start by using M to cover the vertices $V(G_i)$ in G. The completeness of the coordination gadgets guarantees that we find a perfect matching in the d-uniform hypergraph $G' - V(G_i)$ that uses only hyperedges of the coordination gadgets. By Lemma 4.2, this gives rise to an H-factor of G.

For the soundness, assume we have an H-factor M of G. Lemma 4.2 guarantees that the edge gadgets can be seen as p-hyperedges in the intermediate graph G'. Soundness of the coordination gadgets guarantees that M leaves exactly one group free per part. Now let H' be a copy of H that is contained in G but not in any of the gadgets. Since H' has chromatic number p, H' intersects all p parts and has an edge between any two distinct parts. By construction of G, this implies that the projection of H onto P is a clique. By the packing lemma, this clique is one of the K_i's. Therefore, each H' of the H-factor M that is not in one of the gadgets is contained in G_i, which implies that G_i has an H-factor.

The claim follows since G is a graph on $\sqrt{t}^{1+o(1)}\,\mathrm{poly}(s)$ vertices that has an H-factor if and only if some G_i has an H-factor. ∎

Now Lemma 2.1 immediately implies Theorem 1.4, our kernel lower bounds for H-Factor.

5 Kernels for Graph Packing Problems

The sunflower kernelization in Theorem 1.1 immediately transfers to H-Matching for any fixed graph H and yields kernels with $O(k^d)$ edges. For every graph H, Moser [Mos09] shows that H-matching has kernels with $O(k^{d-1})$ *vertices* where $d = |V(H)|$, but this gives only the weaker bound $O(k^{2d-2})$ on the number of edges. Here we show that for some specific H, we can obtain kernels that are better than the $O(k^d)$ bound implied by Theorem 1.1. As a very simple example, we show this first for $K_{1,d}$-Matching, the problem of packing vertex-disjoint stars with d leaves.

Observation 5.1. *$K_{1,d}$-Matching has kernels with $O(k^2)$ edges.*

Proof. Let (G, k) be an instance of $K_{1,d}$-Matching. If G has a vertex v of degree at least $dk + 1$, let e be an edge incident to v. We claim that we can safely remove e. If $G - e$ has a $K_{1,d}$-matching of size k, then this also holds for G. For the other direction, let M be a $K_{1,d}$-matching of size k in G. If M does not contain e, it is also a matching of $G - e$. Otherwise M contains e. Let M' be obtained from M by removing the star that contains e. Now v is not contained in M'. Since M' covers at most $d(k-1)$ vertices, at least $d+1$ neighbors of v are not contained in M'. Even if we remove e, we can therefore augment M' with a vertex-disjoint star that is centered at v and has d leaves. This yields a star matching of size k in $G - e$.

For the kernelization, we repeatedly delete edges incident to high-degree vertices. Then every vertex has degree at most dk. Now we greedily compute a maximal star matching M and answer 'yes' if M has size k. Otherwise, we claim that the graph has most $O(k^2)$ edges: Since M covers at most dk vertices, the degree bound implies that at most $(dk)^2$ edges are incident to M. The vertices of G outside of M have at most $d - 1$ neighbors outside of M because they would otherwise have been added to M. Thus there are at most $(d - 1) \cdot (dk)^2$ edges not incident to M. Thus G has at most $d^3 \cdot k^2$ edges. ■

By Theorem 1.4, it is unlikely that star matching problems have kernels with $O(k^{2-\epsilon})$ edges, so the above kernels are likely to be asymptotically optimal.

5.1 Packing Paths of Length 3.

Let P_ℓ be the simple path with ℓ edges. As P_2 is the same as $K_{1,2}$, the problem P_2-Matching is already covered by Observation 5.1, thus we have a $O(k^2)$ upper bound and a matching $O(k^{2-\epsilon})$ lower bound for this problem. For P_3-Matching, the situation is less clear. Using a similar strategy as in the proof of Observation 5.1, it is easy to reduce the maximum degree to $O(k^2)$ and then argue that the kernels have $O(k^3)$ edges. Surprisingly, the maximum degree can be further reduced to $O(k^{1.5})$ using much more complicated combinatorial arguments. This gives rise to kernels of size $O(k^{2.5})$ without a tight lower bound.

Theorem 5.1. *P_3-Matching has kernels with $O(k^{2.5})$ edges.*

We prove Theorem 5.1 by showing that the degree of every vertex can be reduced to $\Delta \leq O(k^{1.5})$. Once we have an instance G with maximum degree Δ, we can obtain a kernel of size $O(\Delta \cdot k)$ with fairly standard arguments as follows. First, we greedily compute a maximal P_3-matching. If we find at least k paths, then we are done. Otherwise let S be the at most $4k$ vertices in the paths. As every vertex has degree at most Δ, there are at most $4k\Delta$ edges incident to S. Now let us count the number of edges in $G \setminus S$. The graph $G \setminus S$ does not contain paths of length 3, so every connected component of $G \setminus S$ is either a triangle or a star. Therefore, the average degree is at most 2 in $G \setminus S$. If a component of $G \setminus S$ is not adjacent to S, it can be safely removed without changing the solution. If a component of $G \setminus S$ has a vertex v with at least two neighbors in $G \setminus S$ that have degree one in G, then we keep only one of them. Since every solution uses at most one of them, they are interchangeable. After doing this, every component of $G \setminus S$ has at most two vertices not adjacent to S in G. This means that a constant fraction of the vertices in $G \setminus S$ is adjacent to S. As there are at most $4\Delta k$ edges incident to S, this means that there are at most $O(\Delta k)$ vertices in $G \setminus S$. Taking into account that the average degree is at most two in $G \setminus S$, we have that there are $O(\Delta \cdot k)$ edges in $G \setminus S$. This yields kernels with $O(k^{2.5})$ edges. It remains to argue how to reduce the maximum degree to Δ.

Degree reduction. Let G be a graph that contains a vertex v with more than Δ neighbors. In the following, we call any P_3-matching of size k a *solution*. Our kernelization procedure will find an edge e incident to v that can be *safely removed*, so that G has a solution if and only $G \setminus e$ has a solution. The most basic such reduction is as follows.

Lemma 5.1. *If there there is a matching $a_1b_1, \ldots, a_nb_n$ of size $n \geq 4k + 2$ in $G \setminus v$ such that every a_i is a neighbor of v, then any single edge e incident to v can be safely removed.*

Proof. Suppose that there is a solution containing a path going through e. The paths in the solution cover $4k$

vertices, thus without loss of generality, we can assume that a_1, b_1, a_2, b_2 are not used. We replace the path containing e with the path $b_1a_1va_2$ to obtain a solution of $G \setminus e$. ∎

Let us greedily find a maximal matching $a_1b_1, \ldots, a_nb_n$ in $G \setminus v$ with the requirement that every a_i is a neighbor of v. If $n \geq 4k + 2$, we can safely remove an arbitrary edge incident to v by Lemma 5.1 and then proceed inductively. Otherwise, let $M = \{a_1, b_1, \ldots, a_n, b_n\}$ be the set of at most $8k+2$ vertices that are covered by this matching. Let $X := N(v) \setminus M$. Now every neighbor y of a vertex $x \in X$ is in $M \cup \{v\}$ since the matching M could otherwise have been extended by the edge xy. In particular, X induces an independent set. It holds that $|X| \geq 100k$ since otherwise the degree of v is smaller than Δ.

The following technical definition is crucial in our kernelization algorithm.

Definition 5.1. *Let u be a vertex of M and let $X_u = N(u) \cap X$ be the neighborhood of u in X.*

We call u good *if every set $S \subseteq M$ satisfies the following property: If there is a matching between S and X_u of size $|X_u| - 1$, then S has more than $4k$ neighbors in X.*

Note that it is not obvious how to decide in polynomial time whether a vertex is good. Therefore, Lemma 5.2 below does not directly give us a polynomial-time reduction rule. We will invoke it only in situations where we can prove that all the required vertices are good.

Lemma 5.2. *If $x \in X$ has only good neighbors in M, then the edge vx can be safely removed.*

Proof. We argue that if there is a solution then there is also a solution that does not use vx. If vx is used as the first or the third edge of a path, the high degree of v makes sure that there is a vertex y not used by the solution, and we can replace vx by vy. Now consider a solution that contains a path $P = avxu$ using vx as its middle edge; by assumption, $u \in M$ is good.

By definition, the set X_u contains all vertices x' of X that are common neighbors of u and v. Hence, if some vertex $x' \in X_u \setminus x$ is not used by the solution, then we can replace P by $avx'u$. Now assume that every $x' \in X_u \setminus x$ is part of some path. None of these paths contain v. If x' is the endpoint of a path, then the *mate* of x' is its unique neighbor in the path; if x' is in the middle of a path, then the mate of x' is the endpoint that is adjacent to x' in the path. Recall that every neighbor of $x' \in X$ is in $M \cup \{v\}$, so the mate of any x' is contained in M. The vertices in $X_u \setminus x$ have distinct mates even if two vertices of $X_u \setminus x$ are on the same path. This gives rise to a matching between $X_u \setminus x$ and the set $S \subseteq M$ of all mates of vertices in $X_u \setminus x$. Since this matching has size $|X_u| - 1$ and u is good, S has at least $4k+1$ neighbors in X. Thus, some neighbor $y \in X$ of S is not used by the solution.

Let $x' \in X_u \setminus x$ be a vertex whose mate $w \in S$ is adjacent to a vertex $y \in X$ that is not used by the solution. Since w is the mate of x', the edge wx' occurs in a path Q of the solution. We distinguish two cases. If x' is an endpoint of Q, then we replace the paths $P = avxu$ and $Q = x'wcd$ by the two new paths $avx'u$ and $ywcd$. If x' is not an endpoint of Q, then we replace $P = avxu$ and $Q = wx'cd$ by $ux'cd$ and $avyw$. These are paths since x' is a common neighbor of v and u, and y is a common neighbor of v and w. In all cases we found solutions that do not use vx, so vx can be safely removed. ∎

Lemma 5.3. *There is a polynomial-time algorithm that, given a vertex v of degree larger than Δ, finds a vertex $x \in X$ that has only good neighbors in M.*

Proof. We maintain a set $M' \subseteq M$ of vertices satisfying the invariant that all vertices in M' are good. Initially we set $M' = \emptyset$. We repeat a procedure that either outputs x as required or adds a new good vertex to M'. If some $x \in X$ does not have neighbors in $M \setminus M'$, then by the invariant all neighbors of x in M are good and we can output x. Otherwise, with $M \setminus M' = \{m_1, \ldots, m_t\}$, there exists a partition $X^1, \ldots, X^t$ of X such that every vertex of X^i is adjacent to m_i. Some of the X^i can be empty.

We construct a bipartite graph H that is a subgraph of the bipartite graph between X and M. Initially, H has the vertex set $X \cup M$ and no edges. We preserve the invariant that every vertex of X has degree at most one in H.

For every $1 \leq i \leq t$ with $|X^i| > 1$, we add edges to H in the following way. For every edge xy of G with $x \in X^i$ and $y \in M$, let the *weight* of xy be the degree $\deg_H(y)$ of y in H. In this weighted graph G, we now compute a matching between X^i and M that has cardinality exactly $|X^i| - 1$ and weight at most $4k$. This can be done in polynomial time using standard algorithms. If there is such a matching, we add all edges of the matching to H and continue with the next i. This preserves the invariant that every vertex of X has degree at most one in H since the X^i are disjoint. If there is no such matching, then we claim that m_i is good. Assume for contradiction that there is a matching of cardinality $|X_{m_i}| - 1$ between $X_{m_i} = N(m_i) \cap X$ and a subset $S \subseteq M$ that has at most $4k$ neighbors in X. As H is a subgraph of G, it follows that S has at most $4k$

neighbors in H. This implies that $\sum_{y\in S}\deg_H(y) \le 4k$ since every vertex of X has degree at most one in H, so the sum of the degrees of vertices in S is exactly the size of the neighborhood of S in H. This contradicts with the fact that we did not find a suitable matching of weight at most $4k$. Thus m_i is good and can be added to M'.

We show that unless $|X| = O(k^{1.5})$, the above process finds a good vertex in M. Suppose that the process terminates without finding a good vertex. Let N be the number of paths of length two in the final graph H we obtained. As the degree of every vertex of X is at most one in H, every path of length two is of the form abc with $a \in X^i$, $b \in M$, and $c \in X^j$ for some $1 \le i, j \le t$. Furthermore, we have $i \ne j$: the edges incident to X^i form a matching in H. For some i, let us count the number of paths with $a \in X^i$ and $c \in X^j$ for $j < i$. Consider a vertex $a \in X^i$ that is not isolated in H; it has a unique neighbor b in the graph H. Consider the graph H' at the step of the algorithm before finding the matching incident to X^i, and let d be the degree of b in H'. Then it is clear that H contains exactly d paths of length two connecting a to a vertex of X^j with $j < i$: the vertex b has exactly d neighbors in $X^1 \cup \cdots \cup X^{i-1}$. Thus if S_i is the set of vertices that X^i is matched to, then the total number of paths between X^i and $\bigcup_{j=1}^{i-1} X^j$ is exactly the total degree of S_i in H', which is at most $4k$ by the selection of the matching. Thus the total number N of paths can be bounded by $t \cdot 4k \le (8k+2)\cdot 4k = O(k^2)$.

On the other hand, the number of paths of length two containing $m \in M$ as their middle vertex is exactly

$$\binom{\deg_H(m)}{2} \ge \deg_H(m)^2/4 - 1.$$

Note that $\sum_{m\in M}\deg_H(m) \ge |X| - |M|$: in every nonempty X^i, there is exactly one vertex that is isolated in H, and every other vertex has degree one. Thus the total number of paths is exactly

$$\begin{aligned}\sum_{m\in M}\binom{\deg_H(m)}{2} &\ge \frac{1}{4}\sum_{m\in M}\deg_H(m)^2 - |M| \\ &\ge \frac{1}{4|M|}\left(\sum_{m\in M}\deg_H(m)\right)^2 - |M| \\ &\ge \frac{1}{4|M|}(|X|-|M|)^2 - |M| = \Omega(|X|^2/k)\end{aligned}$$

where we used the relationship between arithmetic and quadratic mean in the second inequality, and the facts $|M| \le 8k+2$, $|X| > 100k$ in the last step. Putting together $N = O(k^2)$ and $N = \Omega(|X|^2/k)$, we get $|X| = O(k^{1.5})$.

Thus, we choose $\Delta = C \cdot k^{1.5}$ for some large enough constant $C > 0$ so that the above procedure is guaranteed to find a vertex $x \in X$ that contains only good neighbors in M. ■

Acknowledgements. We would like to thank Martin Grohe and Dieter van Melkebeek for valuable comments on previous versions of this paper.

References

[BDFH09] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.

[BFL$^+$09] Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (meta) kernelization. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009*, pages 629–638, 2009.

[BJK11a] Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Cross-composition: A new technique for kernelization lower bounds. In *Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science, STACS 2011*, volume 9 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 165–176, 2011.

[BJK11b] Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Preprocessing for treewidth: A combinatorial analysis through kernelization. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming, ICALP 2011*, volume 6755 of *Lecture Notes in Computer Science*, pages 437–448. Springer, 2011.

[BTY09] Hans L. Bodlaender, Stéphan Thomassé, and Anders Yeo. Kernel bounds for disjoint cycles and disjoint paths. In *Proceedings of the 17th Annual European Symposium on Algorithms, ESA 2009*, volume 5757 of *Lecture Notes in Computer Science*, pages 635–646. Springer, 2009.

[CFJ04] Benny Chor, Michael R. Fellows, and David W. Juedes. Linear kernels in linear time, or how to save k colors in O(n^2) steps. In *Proceedings of the 30th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2004)*, volume 3353 of *Lecture Notes in Computer Science*, pages 257–269. Springer, 2004.

[CFM11] Yijia Chen, Jörg Flum, and Moritz Müller. Lower bounds for kernelizations and other preprocessing procedures. *Theory of Computing Systems*, 48(4):803–839, 2011.

[DF99] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Monographs in Computer Science. Springer, New York, 1999.

[DLS09] Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Incompressibility through colors and IDs.

In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming, ICALP 2009*, volume 5555 of *Lecture Notes in Computer Science*, pages 378–389. Springer, 2009.

[DvM10] Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proceedings of the 42th Annual ACM Symposium on Theory of Computing, STOC 2010*, pages 251–260. ACM, 2010.

[ER60] Paul Erdős and Richard Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35:85–90, 1960.

[FFL$^+$09] Henning Fernau, Fedor V. Fomin, Daniel Lokshtanov, Daniel Raible, Saket Saurabh, and Yngve Villanger. Kernel(s) for problems with no kernel: On out-trees with many leaves. In *Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science, STACS 2009*, volume 3 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 421–432, 2009.

[FG06] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[FHR$^+$04] Michael R. Fellows, Pinar Heggernes, Frances A. Rosamond, Christian Sloper, and Jan Arne Telle. Finding k disjoint triangles in an arbitrary graph. In *Proceedings of the 30th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2004)*, volume 3353 of *Lecture Notes in Computer Science*, pages 235–244. Springer, 2004.

[FKN$^+$08] Michael R. Fellows, Christian Knauer, Naomi Nishimura, Prabhakar Ragde, Frances A. Rosamond, Ulrike Stege, Dimitrios M. Thilikos, and Sue Whitesides. Faster fixed-parameter tractable algorithms for matching and packing problems. *Algorithmica*, 52:167–176, 2008.

[FLST10] Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Dimitrios M. Thilikos. Bidimensionality and kernels. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010*, pages 503–510. SIAM, 2010.

[FR09] Henning Fernau and Daniel Raible. A parameterized perspective on packing paths of length two. *Journal of Combinatorial Optimization*, 18(4):319–341, 2009.

[FS08] Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC 2008*, pages 133–142. ACM, 2008.

[Guo09] Jiong Guo. A more effective linear kernelization for cluster editing. *Theoretical Computer Science*, 410(8-10):718–726, 2009.

[HW11] Danny Hermelin and Xi Wu. Weak compositions and their applications to polynomial lower-bounds for kernelization. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012*. SIAM, 2011.

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. *Complexity of computer computations*, 43:85–103, 1972.

[KH78] David G. Kirkpatrick and Pavol Hell. On the completeness of a generalized matching problem. In *Proceedings of the 10th annual ACM Symposium on Theory of Computing, STOC 1978*, pages 240–245. ACM, 1978.

[KMW10] Stefan Kratsch, Dániel Marx, and Magnus Wahlström. Parameterized complexity and kernelizability of max ones and exact ones problems. In *Proceedings of the 35th International Symposium on Mathematical Foundations of Computer Science, MFCS 2010*, pages 489–500, 2010.

[KW09] Stefan Kratsch and Magnus Wahlström. Two edge modification problems without polynomial kernels. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation, IWPEC 2009*, volume 5917 of *Lecture Notes in Computer Science*, pages 264–275. Springer, 2009.

[KW10] Stefan Kratsch and Magnus Wahlström. Preprocessing of min ones problems: A dichotomy. In *Proceedings of the 37th Colloquium on Automata, Languages and Programming, ICALP 2010*, volume 6198 of *Lecture Notes in Computer Science*, pages 653–665. Springer, 2010.

[LMS11] Daniel Lokshtanov, Matthias Mnich, and Saket Saurabh. A linear kernel for a planar connected dominating set. *Theoretical Computer Science*, 412(23):2536–2543, 2011.

[Mos09] Hannes Moser. A problem kernelization for graph packing. In *Proceedings of the 35th Conference on Current Trends om Theory and Practice of Computer Science, SOFSEM 2009*, volume 5404 of *Lecture Notes in Computer Science*, pages 401–412. Springer, 2009.

[MPS04] Luke Mathieson, Elena Prieto, and Peter Shaw. Packing edge disjoint triangles: A parameterized view. In *Proceedings of the First International Workshop on Parameterized and Exact Computation, IWPEC 2004*, volume 3162 of *Lecture Notes in Computer Science*, pages 127–137. Springer, 2004.

[MRS11] Neeldhara Misra, Venkatesh Raman, and Saket Saurabh. Lower bounds on kernelization. *Discrete Optimization*, 8(1):110–128, 2011.

[Nie06] Rolf Niedermeier. *Invitation to fixed-parameter algorithms*, volume 31 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2006.

[NT74] George L. Nemhauser and Leslie E. Trotter Jr. Properties of vertex packing and independence system polyhedra. *Mathematical Programming*, 6(1):48–61, 1974.

[PS06] Elena Prieto and Christian Sloper. Looking at the stars. *Theoretical Computer Science*, 351(3):437–445, 2006.

[Tho10] Stéphan Thomassé. A 4^2 kernel for feedback vertex set. *ACM Transactions on Algorithms*, 6(2), 2010.

[WNFC10] Jianxin Wang, Dan Ning, Qilong Feng, and Jianer Chen. An improved kernelization for P_2-

packing. *Information Processing Letters*, 110(5):188–192, 2010.

[Yus07] Raphael Yuster. Combinatorial and computational aspects of graph packing and graph decomposition. *Computer Science Review*, 1(1):12–26, 2007.

A Sunflower Kernelization for Set Matching

We sketch a modern proof of Theorem 1.1, that d-Set Matching has kernels with $O(k^d)$ hyperedges.

Proof (Sketch). A *sunflower* with p petals is a set of p hyperedges whose pairwise intersections are equal. By the sunflower lemma, any d-uniform hypergraph G with more than $d! \cdot r^d$ edges has a sunflower with $r+1$ petals [ER60]. We set $r = dk$ and observe that, in any sunflower with $r+1$ petals, we can arbitrarily choose an edge e of the sunflower and remove it from the graph. To see this, assume we have a matching M of G with k edges. If M does not contain e, then M is still a matching of size k in $G - e$. On the other hand, if M contains e, there must be a petal that does not intersect M since we have $dk+1$ petals but M involves only dk vertices. Thus we can replace e in the matching by the edge that corresponds to that petal, and we obtain a matching of G' that consists of k hyperedges. This establishes the completeness of the reduction. The soundness is clear since any matching of G' is a matching of G. ■

B Multicolored Biclique

Lemma B.1. Multicolored Biclique *is* NP-*complete.*

Proof. Let graph G and integer k be an instance of Clique. Let $\{v_i \mid 1 \leq i \leq n\}$ be the vertex set of G. We construct a biparite graph B on vertex set $\{u_{i,j}, w_{i,j} \mid 1 \leq i \leq k, 1 \leq j \leq n\}$. We make vertices $u_{i,j}$ and $v_{i',j'}$ adjacent if and only if

- either $i = i'$ and $j = j'$ or
- $i \neq i'$ and vertices v_j and $v_{j'}$ are adjacent.

Consider the partitions $U = U_1 \cup \cdots \cup U_k$ and $W = W_1 \cup \cdots \cup W_k$, where $U_i = \{u_{i,j} \mid 1 \leq j \leq n\}$ and $W_i = \{w_{i,j} \mid 1 \leq j \leq n\}$. We claim that B contains a biclique $K_{n,n}$ respecting these partitions if and only if G contains a k-clique. It is easy to see that if $\{v_{a_1}, \ldots, v_{a_k}\}$ is a clique in G, then $\{u_{1,a_1}, \ldots, u_{k,a_k}, w_{1,a_1}, \ldots, w_{k,a_k}\}$ is a biclique of the required form in B. On the other hand, if $\{u_{1,a_1}, \ldots, u_{k,a_k}, w_{1,b_1}, \ldots, w_{k,b_k}\}$ is such a biclique, then $a_i = b_i$ for every $1 \leq i \leq k$; otherwise u_{i,a_i} and w_{i,b_i} are not adjacent. It follows that $\{v_{a_1}, \ldots, v_{a_k}\}$ is a clique in G: if v_{a_i} and $v_{a_{i'}}$ are not adjacent in G (including the possibility that $a_i = a_{i'}$), then u_{i,a_i} and $w_{i',b_{i'}} = w_{i',a_{i'}}$ are not adjacent in B. ■

C Lower Bounds for Vertex Cover in d-uniform Hypergraphs

We present an elementary reduction from OR(3-Sat) to d-Vertex Cover, i.e., the vertex cover problem in d-uniform hypergraphs. The d-partiteness flavor is crucial in the reduction, but it is not necessary to explicitly spell out the d-partite problem L like we did with Multicolored Biclique before.

Theorem C.1 ([DvM10]). *Let $d \geq 2$ be an integer. Then d-*Vertex Cover *does not have kernels of size $O(k^{d-\epsilon})$ unless* coNP $\subseteq$ NP/poly.

Proof. Let $\varphi_1, \ldots, \varphi_t$ be t instances of 3-Sat, each of size s. Without loss of generality, assume that the set of variables occurring in the formulas is a subset of $[s]$. Let P be the consistency graph on partial assignments that assign exactly three variables of $[s]$. More precisely, the vertex set of P is the set of functions $\sigma : S \to \{0,1\}$ for sets $S \in \binom{[s]}{3}$, and two partial assignment $\sigma, \sigma' \in V(P)$ are adjacent in P if and only if σ and σ' are consistent, i.e., they agree on the intersection of their domains. Now the cliques of size $\binom{s}{3}$ in P are exactly the cliques that are obtained from full assignments $[s] \to \{0,1\}$ by restriction to their three-variable sub-assignments. Furthermore, P has no clique of size larger than $\binom{s}{3}$.

We construct a hypergraph G that has a complete d-uniform sub-hypergraph on some number k of vertices if and only if some φ_i is satisfiable. We use a suitable bijection between $[t]$ and $[t^{1/d}]^d$, and we write the φ_i's as $\varphi_{b_1,\ldots,b_d}$ for $(b_1, \ldots, b_d) \in [t^{1/d}]^d$. The vertex set of G consists of $d \cdot t^{1/d}$ groups of vertices $V_{a,b}$ for $a \in [d]$ and $b \in [t^{1/d}]$. We consider each set $V_{1,b}$ as a copy of the vertex set of P, and for $a > 1$, we let $|V_{a,b}| = 1$ for all b. A subset e of d elements of $V(G)$ is a hyperedge in G if and only if the following properties hold:

1. each $a \in [d]$ has at most one $b = b_a \in [t^{1/d}]$ for which $e \cap V_{a,b} \neq \emptyset$,
2. $e \cap V_{1,b_1}$ corresponds to a clique in P, and
3. if $e \cap V_{a,b_a} \neq \emptyset$ for all a, then the (unique) partial assignment $\sigma \in e \cap V_1$ does not set any clause of $\varphi_{b_1,\ldots,b_d}$ to false.

Edges with $|e \cap V_{1,b_1}| > 1$ play the role of checking the consistency of partial assignments, and edges with $|e \cap V_{a,b_a}| = 1$ for all a select an instance φ_b and check whether that instance is satisfiable.

We set $k = \binom{s}{3} + d - 1$. For the completeness of the reduction, let $\sigma : [s] \to \{0,1\}$ be a satisfying assignment of $\varphi_{b_1,\ldots,b_d}$. Let C be the set of all three-variable sub-assignments of σ in the set $V_{1.b_1}$, and we also add the

$d-1$ vertices of $V_{2,b_2} \cup \cdots \cup V_{d,b_d}$ to C. We claim that C induces a clique in G. Let e be a d-element subset of C, we show that it is a hyperedge in G since it satisfies the three conditions above. Clearly, $e \subseteq C$ is fully contained in $V_{1,b_1} \cup \cdots \cup V_{d,b_d}$ and satisfies the first condition. The second condition is satisfied since $e \cap V_{1,b_1}$ contains only sub-assignments of the full assignment σ. The third condition holds since σ is a satisfying assignment and therefore none of its sub-assignments sets any clause to false.

For the soundness, let C be a clique of size k in G. By the first property, C intersects at most one set V_{a,b_a} for all a. Also, the intersection $C \cap V_{1,b_1}$ induces a clique in G and therefore corresponds to a clique of P. By the properties of P, this intersection can have size at most $\binom{s}{3}$, and the only other vertices C can contain are the $d-1$ vertices of $V_{2,b_2} \cup \cdots \cup V_{d,b_d}$. Thus, we indeed have $|C \cap V_{1,b_1}| = \binom{s}{3}$ and $|C \cap V_{2,b_2}| = \cdots = |C \cap V_{d,b_d}| = 1$. The properties of P imply that the first intersection corresponds to some full assignment $\sigma : [s] \to \{0,1\}$. By the third property, no three-variable sub-assignment sets any clause of $\varphi_{b_1,\ldots,b_d}$ to false, so σ satisfies the formula.

Thus, $(G,k) \in d$-Clique if and only if $(\varphi_1,\ldots,\varphi_t) \in$ OR(3-Sat). Since G and k are computable in time polynomial in the bitlength of $(\varphi_1,\ldots,\varphi_t)$ and $|V(G)| \leq t^{1/d} \cdot \mathrm{poly}(s)$, we have established the $\leq_m^p$-reductions that are required to apply Lemma 2.1. ∎