A Parameterized Perspective on Packing Paths of Length Two

Henning Fernau[1] Daniel Raible[1]

Universität Trier, FB IV—Abteilung Informatik, 54286 Trier, Germany,
{fernau,raible}@informatik.uni-trier.de

Abstract. We study (vertex-disjoint) packings of paths of length two (i.e., of P_2's) in graphs under a parameterized perspective. Starting from a maximal P_2-packing $\mathcal{P}$ of size j we use extremal combinatorial arguments for determining how many vertices of $\mathcal{P}$ appear in some P_2-packing of size $(j+1)$ (if it exists). We prove that one can 'reuse' $2.5j$ vertices. Based on a WIN-WIN approach, we build an algorithm which decides if a P_2-packing of size at least k exists in a given graph in time $\mathcal{O}^*(2.482^{3k})$.

1 Introduction and Definitions

Mathematical Motivation. We consider a natural generalization of the well-known matching problem in graphs. Recall that a maximum matching is a maximum cardinality set of vertex disjoint edges, i.e., a packing with paths of length one. We are going to study packings by paths of length two (abbreviated as P_2). More formally, we consider the following problem, called P_2-PACKING:

Given: A graph $G = (V, E)$, and the parameter k.
We ask: Is there a set of k vertex-disjoint P_2's in G?

P. Hell and D. Kirkpatrick [11, 9] proved $\mathcal{NP}$-completeness for this problem. In fact, they showed that general MAXIMUM H-PACKING is $\mathcal{NP}$-complete. Here, H is a graph with at least three vertices in some connected component. Notice that P_2-PACKING attracts attention as it is $\mathcal{NP}$-hard, whereas the classical matching problem, which is P_1-PACKING, is solvable in polynomial time.

Parameterized interests. H. Fernau and D. F. Manlove [6] discovered a primal-dual relation to TOTAL EDGE COVER. Recall that an *edge cover* is a set of edges $EC \subseteq E$ that cover all vertices of a given graph $G = (V, E)$. An edge cover is called *total* if every component in $G[EC]$ has at least two edges. This type of constraint for covering problems is motivated by modelling clustering properties within cover sets, see [6]. By matching techniques, the problem of finding an edge cover of size at most k is solvable in polynomial time. However, the following Gallai-type identity [6] proves that finding total edge covers of size at most k is $\mathcal{NP}$-hard: The sum of the number of P_2's in a maximum P_2-packing and the size of a minimum total edge cover equals $n = |V|$. H. Fernau and D. F. Manlove [6] also showed that TOTAL EDGE COVER is fixed-parameter tractable (or: lies in $\mathcal{FPT}$, for short). This is quite interesting since there are few natural, unrestricted problems where both the primal and the dual variant are known to lie in $\mathcal{FPT}$.

Applications. There is a strong link to the TEST COVER (TC) problem [1] with applications ranging from fault testing and diagnosis, pattern recognition to biological identification. The input to TC is a hypergraph $H = (G, E)$ and one wishes to identify a subset $E' \subseteq E$ (the *test cover*) such that, for any distinct $i, j \in V$, there is an $e' \in E'$ with $|e' \cap \{i.j\}| = 1$. TC models identification problems: Given a set of individuals and a set of binary attributes we search for a minimum subset of attributes that identifies each individual distinctly. For the special yet important case TCP2, where for all $e \in E$ we have $|e| \leq 2$, K. M. J. Bontridder *et al.* [1] could show the following two assertions. (1) If H has a test cover of size τ, then there is a P_2-packing of size $n - \tau - 1$ that leaves at least one vertex isolated. (2) If H has a maximal P_2-packing of size π that leaves at least one vertex isolated, then there is a test cover of size $n - \pi - 1$. This also establishes a close relation between TEST COVER and TOTAL EDGE COVER. So we can employ our algorithms to solve the TCP2 case of TEST COVER by using an initial catalytic branch that determines one vertex that should be isolated.

Discussion of Related Work. R. Hassin and S. Rubinstein [8] found a randomized $\frac{35}{67}$-approximation for finding a maximum P_2-packing. K. M. J. Bontridder *et al.* [1] studied deterministic approximation algorithms, considering a series of heuristics H_ℓ. H_ℓ starts from a maximal P_2-packing $\mathcal{P}$ and tries to improve it by replacing ℓ P_2's by $\ell + 1$ P_2's. The corresponding approximation ratios ρ_ℓ are as follows: $\rho_0 = \frac{1}{3}$, $\rho_1 = \frac{1}{2}$, $\rho_2 = \frac{5}{9}, \rho_3 = \frac{7}{11}$ and $\rho_\ell = \frac{2}{3}$ for $\ell \geq 4$.
As any P_2-PACKING instance can be transformed into a 3-SET PACKING instance one can use Y. Liu *et al.* [12] algorithm which needs $\mathcal{O}^*(4.61^{3k})$ steps, or the very recent algorithm of J. Wang and Q. Feng [14] running in time $\mathcal{O}(3.52^{3k})$. This is the culmination point of a sequence of papers subsequently improving on the running time of this problem. Alternatively, we can use randomized parameterized algorithms; the best published algorithms yields a running time of $\mathcal{O}^*(2.52^{3k})$, see [3]. Recently, we were informed by I. Koutis[1] that he has developed a randomized parameterized algorithm for this problem that runs in time $\mathcal{O}^*(2^{3k})$. The first paper to individually study P_2-PACKING under a parameterized view was E. Prieto and C. Sloper [13]. The authors were able to prove a $15k$-kernel. Via a clever midpoint search on the kernel they could achieve a deterministic run time of $\mathcal{O}^*(3.403^{3k})$. Another special case of 3-SET PACKING studied from a parameterized perspective is 3-DIMENSIONAL MATCHING, see [12] for a deterministic algorithm of run time $\mathcal{O}^*(2.77^{3k})$. Recently, J. Wang *et al.* [15] found a kernel of size $7k$ for P_2-PACKING, resulting in a deterministic $\mathcal{O}^*(2.61^{3k})$-algorithm for this problem.

Our Contributions. The main achievements of this paper are: (1) We present an algorithm which solves this problem in time $\mathcal{O}^*(2.482^{3k})$. (2) We exhibit an

[1] personal communication; the corresponding paper will be presented at ICALP 2008

extremal combinatorial argument to show that, given a P_2-packing of size j and provided that a larger packing exists, we can reuse $2.5j$ vertices of the known packing. This improves a similar result for general 3-SET PACKING [12] where only $2j$ elements are reusable. (3) Another novelty is that in this algorithm, the inductive augmentation step is interleaved with kernelization. This pays off not only heuristically but also asymptotically by a specific form of combinatorial analysis. Thereby we can completely skip the time consuming color-coding which was needed in Liu *et al.* [12] for 3-SET PACKING. (4) We show that WIN-WIN games can be played with two different brute-force algorithms to finally achieve the claimed running time. We believe that especially the idea of saving colors by extremal combinatorial arguments could be applied in other situations, as well.

Some Notations and Definitions. We only consider undirected graphs $G = (V, E)$. For a subgraph H of G, denote by $N(H)$ the set of vertices that are not in H but adjacent to at least one vertex on H, i.e., $N(H) = (\bigcup_{v \in H} N(\{v\})) \setminus H$. The subgraph H is *adjacent* to a vertex v if $v \in N(H)$. A P_2 in G is a path which consists of three vertices and two edges. For any path p of this kind we consider the vertices as numbered such that $p = p_1p_2p_3$ (where the roles of p_1 and p_3 might be interchanged). For a path p, $V(p)$ ($E(p)$, resp.) denotes the set of vertices (edges, resp.) on p. Likewise, for a set of paths $\mathcal{P}$, $V(\mathcal{P}) := \bigcup_{p \in \mathcal{P}} V(p)$ ($E(\mathcal{P}) := \bigcup_{p \in \mathcal{P}} E(p)$, resp.).

Kernelization Based on the work [13] of E. Prieto and C. Sloper, Wang *et al.* exhibibited the following result [15]:

Theorem 1. P_2-PACKING *admits a kernel with at most* $7k$ *vertices.*

That result was obtained by optimizing the use of fat and double crowns through local improvements, called **Rule 1** and **Rule 2**.

We mention here that the (more general results) of H. Fernau and D. Manlove [6] can be improved for the parametric dual (in the sense of the mentioned Gallai-type identity) TOTAL EDGE COVER, parameterized by k_d upperbounding the edge cover size:

Theorem 2. TOTAL EDGE COVER *admits a kernel with at most* $1.5k_d$ *vertices.*

Proof. Since we aim at a total edge cover, the largest number of vertices that can be covered by k edges is $1.5k$ (namely, if the edge cover is a P_2-packing). Hence, if the graph contains more than $1.5k$ vertices, we can reject. This leaves us with a kernel with at most $1.5k$ vertices. □

This also allows us to state lower bounds for the kernel sizes, based on works of J. Chen *et al.* [2]:

Corollary 1. *Trivially,* P_2-PACKING *does not admit a kernel with less than* $3k$ *vertices.* TOTAL EDGE COVER *does not admit a kernel with less than* $\alpha_a k_d$ *vertices for any* $\alpha_d < (7/6)$*, unless* $\mathcal{P} = \mathcal{NP}$.

Proof. A P_2-packing of size k is only possible in a graph with at least $3k$ vertices. Due to Theorem 1 and [2, Theorem 3.1], there does not exist a kernel of size $\alpha_d k_d$ for TOTAL EDGE COVER under the assumption that $\mathcal{P} = \mathcal{NP}$ if $(7-1)(\alpha_d - 1) < 1$. □

2 Combinatorial Properties of P_2-Packings

This section is devoted to proving the following combinatorial result by extremal combinatorial arguments. Notice that $\mathfrak{Q}_{(2)}$ denotes a set of P_2-packings of size $(j+1)$. The exact definition of $\mathfrak{Q}_{(2)}$ will be given later.

Theorem 3. *Let $\mathcal{P}$ be a maximal P_2-packing of size j. If there is a P_2-packing of size $(j+1)$, then there is also a packing $\mathcal{Q} \in \mathfrak{Q}_{(2)}$ with $|V(\mathcal{P}) \cap V(\mathcal{Q})| \geq 2.5j$.*

The combinatorial properties of $\mathcal{Q}$ will be used in the next section by the inductive step of our algorithm for P_2-PACKING. Among all maximal P_2-packings of size $(j+1)$, we will consider those packings $\mathcal{Q}$ that maximize

$$\sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} 1_{[E(p)=E(q)]}, \tag{1}$$

where $1_{[\ ]}$ is the indicator function. We call the set of these packings $\mathfrak{Q}_{(1)}$. In $\mathfrak{Q}_{(1)}$, we find those packings $\mathcal{Q}$ that 'reuse' the maximum number of P_2's from the packing $\mathcal{P}$. From Liu et *al.* [12], we know:

Lemma 1. $|V(p) \cap V(\mathcal{Q})| \geq 2$ *for any $p \in \mathcal{P}$ and $\mathcal{Q} \in \mathfrak{Q}_{(1)}$.*

Proof. If there is $p \in \mathcal{P}$ with $|V(p) \cap V(\mathcal{Q})| = 1$, then replace the intersecting path of $\mathcal{Q}$ by p . In the case where $|V(p) \cap V(\mathcal{Q})| = 0$, simply replace an arbitrary $q \in \mathcal{Q} \backslash \mathcal{P}$, that must exist by pigeon-hole, by p. In both cases, we obtain a packing $\mathcal{Q}'$ of the same size as $\mathcal{Q}$, but $\sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}'} 1_{[E(p)=E(q)]} = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} 1_{[E(p)=E(q)]} + 1$, contradicting $\mathcal{Q} \in \mathfrak{Q}_{(1)}$. □

A slightly sharper version is the next assertion:

Corollary 2. *If $\mathcal{Q} \in \mathfrak{Q}_{(1)}$, then for any $p \in \mathcal{P}$ with $p \notin \mathcal{Q}$, there are $q_1, q_2 \in \mathcal{Q}$, $q_1 \neq q_2$, with $|V(p) \cap V(q_i)| \geq 1$ $(i = 1, 2)$.*

Proof. Suppose it exists $p \in \mathcal{P}$ and only one $q \in \mathcal{Q}$ with $|V(p) \cap V(q)| \geq 2$. Then $\mathcal{Q} \setminus \{q\} \cup \{p\}$ improves on priority (1), contradicting $\mathcal{Q} \in \mathfrak{Q}_{(1)}$. □

We sharpen this combinatorial bound by considering from the set $\mathfrak{Q}_{(1)}$ only those P_2-packings $\mathcal{Q}'$ which maximize the following second property:

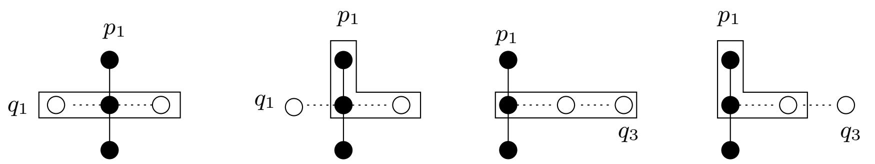
$$\sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}'} |E(p) \cap E(q)|. \tag{2}$$

The set of the remaining P_2-packings will be called $\mathfrak{Q}_{(2)}$. So, in $\mathfrak{Q}_{(2)}$ are those packings from $\mathfrak{Q}_{(1)}$ which cover the maximum number of edges in $E(\mathcal{P})$.

In contrast to the general situation with 3-SET PACKING, paths are more concrete objects that can be shifted or folded along the given graph. These geometric ideas will be used to finally prove our claimed combinatorial theorem.

We define $\mathcal{P}_i(\mathcal{Q}) := \{p \in \mathcal{P} \mid i = |p \cap V(\mathcal{Q})|\}$. A vertex $v \in V$ is a *$\mathcal{Q}$-endpoint* if there is a unique $q = q_1 \dots q_3 \in Q$ such that $v = q_1$ or $v = q_3$. A vertex v is called *$\mathcal{Q}$-midpoint* if there is a $q = q_1q_2q_3 \in \mathcal{Q}$ with $q_2 = v$.

1. We call $q = q_1q_2q_3 \in \mathcal{Q}$ *foldable* on $p = p_1p_2p_3 \in \mathcal{P}$ if, for $q_2 \in V(p) \cap V(q)$, we have $p_s = q_2$, $s \in \{1, 2, 3\}$, and either $p_{s+1} \notin V(\mathcal{Q})$ or $p_{s-1} \notin V(\mathcal{Q})$, see Figure 1(a).
2. If q is foldable on p, then substituting q by $q \setminus \{q_i\} \cup \{p_{s\pm1}\}$ with $i \in \{1, 3\}$, will be called *$(q_i, p_{s\pm1})$-folding*, see Figure 1(b).
3. We call $q = q_1q_2q_3 \in \mathcal{Q}$ *shiftable* with respect to q_1 (q_3, resp.) on $p = p_1p_2p_3 \in \mathcal{P}$ if the following holds: $q_1 \in V(p) \cap V(q)$ ($q_3 \in V(p) \cap V(q)$, resp.) and either $p_{s+1} \notin V(\mathcal{Q})$ or $p_{s-1} \notin V(\mathcal{Q})$ where $p_s = q_1$ ($p_s = q_3$, resp.) and $s \in \{1, 2, 3\}$, see Figure 1(c).
4. If q is shiftable on p with respect to $t \in \{q_1, q_3\}$, then substituting q by $q \setminus \{g\} \cup \{p_{s+1}\}$ (or by $q \setminus \{g\} \cup \{p_{s-1}\}$, resp.), $g \in \{q_1, q_3\} \setminus \{t\}$, will be called *$(g, p_{s+1})$-shifting* (*$(g, p_{s-1})$-shifting*, resp.), see Figure 1(d).



(a) q is foldable on p. (b) (q_1, p_1)-folding. (c) q is shiftable on p. (d) (q_3, p_1)-shifting.

Fig. 1. The black vertices and solid edges indicate the P_2-packing $\mathcal{P}$. The polygons contain the P_2's of the packing $\mathcal{Q}$.

Lemma 2. *If $q = q_1q_2q_3 \in \mathcal{Q}$ with $\mathcal{Q} \in \mathfrak{Q}_{(2)}$ is shiftable on $p \in \mathcal{P}$ with respect to q_1 (or q_3, resp.), then there is some $p' \in \mathcal{P}$ with $p' \neq p$ such that $\{q_3, q_2\} \in E(p')$ (or $\{q_2, q_1\} \in E(p')$, resp.).*

Proof. We examine the case where $V(p) \cap V(q) = \{q_1\}$ and, w.l.o.g., $p_{s+1} \notin V(\mathcal{Q})$. Now assume the contrary. Then by (q_3, p_{s+1})-shifting, we obtain a P_2-packing $\mathcal{Q}'$. Comparing $\mathcal{Q}$ and $\mathcal{Q}'$ with respect to priority 1, $\mathcal{Q}'$ is no worse than $\mathcal{Q}$. But $\mathcal{Q}'$ improves on priority 2, as we gain $\{p_s, p_{s+1}\}$. But this contradicts $\mathcal{Q} \in \mathfrak{Q}_{(2)}$. □

Lemma 3. *If $\mathcal{Q} \in \mathfrak{Q}_{(2)}$, then no $q \in \mathcal{Q}$ is foldable.*

Proof. Suppose some $q \in \mathcal{Q}$ is foldable on p and, w.l.o.g., $p_{s+1} \notin V(\mathcal{Q})$. Then by (q_1, p_{s+1})-folding q we could improve on priority 2 (without weakening priority 1), contradicting $\mathcal{Q} \in \mathfrak{Q}_{(2)}$. $\square$

Suppose there is a path p with $|V(p) \cap V(\mathcal{Q})| = 2$. Then p shares exactly one vertex $p_{q'}, p_{q''}$ with paths $q', q'' \in \mathcal{Q}$ due to Corollary 2. In the following $p_{q'}$ and $p_{q''}$ will always refer to the two cut vertices of the paths $q', q'' \in \mathcal{Q}$ which cut a path p with $|V(p) \cap V(\mathcal{Q})| = 2$.

Lemma 4. *Let $\mathcal{Q} \in \mathfrak{Q}_{(2)}$. Consider $p \in \mathcal{P}$ with $|V(p) \cap V(\mathcal{Q})| = 2$ and neither $p_{q'}$ nor $p_{q''}$ are $\mathcal{Q}$-endpoints. Then one of q', q'' is foldable.*

Proof. Let $i, j \in \{1, 2, 3\}$ such that $p_{q'} = p_i$ and $p_{q''} = p_j$. Then for $f \in \{1, 2, 3\} \setminus \{i, j\}$, we have $p_f \notin V(\mathcal{Q})$. W.l.o.g., $\{p_i, p_f\} \in E(p)$. Then q' is (q'_1, p_f)-foldable. $\square$

Corollary 3. *Let $\mathcal{Q} \in \mathfrak{Q}_{(2)}$ and $p \in \mathcal{P}$ with $|V(p) \cap V(\mathcal{Q})| = 2$. Then one of $p_{q'}, p_{q''}$ must be a $\mathcal{Q}$-endpoint.*

Proof. Assume the contrary. Then using Lemmas 3 and 4 lead to a contradiction. $\square$

Proof. (of Theorem 3) Suppose there is a path $p \in \mathcal{P}$ with $|V(p) \cap V(\mathcal{Q})| = 2$. By Corollary 3, w.l.o.g.. $p_{q'}$ is a $\mathcal{Q}$-endpoint. For $p_{q''}$ there are two possibilities: **a)** $p_{q''}$ is also a $\mathcal{Q}$-endpoint. Let $\{p_f\} = V(p) \setminus \{p_{q'}, p_{q''}\}$. Then, w.l.o.g., $\{p_{q'}, p_f\} \in E(p)$. Therefore $p_{q'}$ is shiftable. **b)** $p_{q''}$ is a $\mathcal{Q}$-midpoint.
Claim. $p_{q''} \neq p_2$: Suppose the contrary. Then w.l.o.g., $p_{q'} = p_1$ and thus q'' is foldable on p by a (q''_1, p_3)-folding. This contradicts Lemma 3. The claim follows. W.l.o.g., we assume $p_{q''} = p_1$. Then it follows that $p_{q'} = p_2$, as otherwise a (q''_1, p_2)-folding would contradict Lemma 3 again. From $p_{q'} = p_2$ and $p_3 \notin V(\mathcal{Q})$ we can derive that also in this case $p_{q'}$ is shiftable.

We now examine for both cases the implications of the shiftability of $p_{q'}$. W.l.o.g., we suppose that $p_{q'} = q'_1$. Due to Lemma 2 there is a $p' \in \mathcal{P}$ with $\{q'_3, q'_2\} \in E(p')$. From Corollary 2, it follows that there must be a $\bar{q} \in \mathcal{Q} \setminus \{q'\}$ with $|V(p') \cap V(\bar{q})| = 1$. Hence, $|V(p') \cap V(\mathcal{Q})| = 3$. Note that q' is the only path in $\mathcal{Q}$ with $|V(q') \cap V(p')| = 2$. Summarizing, we can say that for any $p \in \mathcal{P}$ with $|V(p) \cap V(\mathcal{Q})| = 2$ we find a distinct $p' \in \mathcal{P}$ (via q') such that $|V(p') \cap V(\mathcal{Q})| = 3$. So, there is a total injection γ from $\mathcal{P}_2(\mathcal{Q})$ to $\mathcal{P}_3(\mathcal{Q})$. From $|\mathcal{P}_2(\mathcal{Q}) \cup \mathcal{P}_3(\mathcal{Q})| = j$ and the existence of γ we derive $|\mathcal{P}_2(\mathcal{Q})| \leq 0.5j$. This implies $|V(\mathcal{P}) \cap V(\mathcal{Q})| = 2|\mathcal{P}_2(\mathcal{Q})| + 3|\mathcal{P}_3(\mathcal{Q})| \geq 2.5j$. $\square$

3 The Algorithm

We are going to discuss three main aspects of Algorithm 1: (1) how matching techniques can be used in the WIN-WIN-approach, (2) why the algorithm is yielding a correct solution, and (3) how the run time is estimated.

Algorithm 1 An Algorithm for finding a P_2-packing $\mathcal{P}$ with $|\mathcal{P}| \geq k$ if possible.

1: **repeat**
2:   {Apply the crown-based kernelization algorithm exhibited in [15].}
3:   Greedily extend $\mathcal{P}$ to a maximal packing using **Rule 1** and **Rule 2**.
4:   Try to find a double or fat crown.
5: **until** $\mathcal{P}$ is not changed
6: $j \leftarrow |\mathcal{P}|$.
7: **if** $j \geq k$ **then**
8:   **return** YES
9: $\mathcal{P}' \leftarrow \emptyset$
10: **for** ℓ=0 to $0.3251j$ **do**
11:   **for all** $S_i \subseteq V(\mathcal{P})$, $S_o \subseteq V \setminus V(\mathcal{P})$ with $|S_i| = (j+1) - \ell$ and $|S_o| = \ell$ **do**
12:     Try to construct a P_2-packing $\mathcal{P}'$ with $S_i \cup S_o$ as midpoints.
13: **for** $\bar{\ell} = 0$ to $0.1749j + 3$ **do**
14:   **for all** $B_i \subseteq V(\mathcal{P})$, $B_o \subseteq V \setminus V(\mathcal{P})$ with $|B_i| = 2(j+1) - \bar{\ell}$ and $|B_o| = \bar{\ell}$ **do**
15:     **for all** possible endpoint pairs $(e_1^1, e_2^1), \ldots, (e_1^{j+1}, e_2^{j+1})$ from $B_i \cup B_o$ **do**
16:       Try to construct a P_2-packing $\mathcal{P}'$with $(e_1^1, e_2^1), \ldots, (e_1^{j+1}, e_2^{j+1})$ as endpoint pairs.
17: **if** $\mathcal{P}' \neq \emptyset$ **then**
18:   $\mathcal{P} \leftarrow \mathcal{P}'$; goto 1.
19: **else**
20:   **return** NO

3.1 Used Matching Techniques

We would like to point out the following two facts about P_2-packings. First, if a graph has a P_2-packing $\mathcal{P} = \{p^1, \ldots, p^k\}$, then it suffices to know the set of midpoints $\mathcal{M}_{\mathcal{P}} = \{p_2^1, \ldots, p_2^k\}$ to construct a P_2-packing of size k (which is possibly $\mathcal{P}$) in polynomial time. This fact was discovered by E. Prieto and C. Sloper [13] and basically can be achieved by bipartite matching techniques. Secondly, it also suffices to know the set of endpoint pairs $E_{\mathcal{P}} = \{(p_1^1, p_3^1), \ldots, (p_1^k, p_3^k)\}$ to construct a P_2-packing of size k in polynomial time. This is due to Lemma 3.3 of Jia *et al.* [10] as any P_2-packing instance also can be viewed as a 3-SET PACKING instance. This is the basic ingredient for the WIN-WIN strategy used in Alg. 1 to finally tune the running time.

Details on the mentioned matching techniques can be found in the following two propositions.

Proposition 1. *Let the vertex set $\mathcal{M} = \{m_1, \ldots, m_j\}$ contain all the midpoints of some P_2-packing $\mathcal{P}$ in a graph $G(V, E)$. Then we can construct a P_2-packing $\mathcal{P}'$ of size j in polynomial time.*

Proof. Use the following algorithm:

– Find a maximum matching M in the auxiliary bipartite graph $G' = (V', E')$, where $V' = A \cup B$ is the bipartition with $A = \mathcal{M} \times \{1, 2\}$ and $B = V \setminus \mathcal{M}$, $E' = \{\{(u, i), w\} \mid 1 \leq i \leq 2, u \in A, w \in B, \{u, w\} \in E\}$.

- If all elements of A are matched in M, then we have found a packing $\mathcal{P}'$ of G as follows: $\mathcal{P}' = \{(x, y, z) \mid \{\{(y,1), x\}, \{(y,2), z\}\} \subseteq M\}$.

Note that $M_{\mathcal{P}} = \{\{\{(p_2,1), p_3\}, \{(p_2,2), p_1\}\} \mid p_1p_2p_3 \in \mathcal{P}\}$ matches A into B in G'. Thus, $\mathcal{P}'$ must exist and is of size j. □

Proposition 2. *Let the tuple set* $\mathcal{E} = \{(e_1^1, e_2^1), \ldots, (e_1^j, e_2^j)\}$ *contain all endpoint pairs of some* P_2*-packing* $\mathcal{P}$ *in* $G(V, E)$*. Then we can construct a* P_2*-packing* $\mathcal{P}'$ *of size* j *in polynomial time.*

Proof. Use the following algorithm:

- Find a maximum matching M in the auxiliary bipartite graph $G' = (V', E')$, where $V' = A \cup B$ is the bipartition with $A = \mathcal{E}$ and $B = V \setminus \{v \in V \mid \exists (e_1^h, e_2^h) \in \mathcal{E}$ with $v = e_1^h$ or $v = e_2^h\}$, $E' = \{\{(e_1^h, e_2^h), u\} \mid (e_1^h, e_2^h) \in A, u \in B, \{e_1^h, u\} \in E$ and $\{e_2^h, u\} \in E\}$
- If all elements of A are matched in M, then we have found a packing $\mathcal{P}'$ of G as follows: $\mathcal{P}' = \{(e_1^h, e_2^h, u) \mid \{(e_1^h, e_2^h), u\} \in M\}$

Note that $M_{\mathcal{P}} = \{\{(p_1, p_3), p_2\} \mid p_1p_2p_3 \in \mathcal{P}\}$ matches A into B in G'. Thus, $\mathcal{P}'$ must exist and is of size j. □

3.2 Correctness

The correctness of the kernelization part is shown in [15].

If a P_2-packing $\mathcal{P}'$ with $|\mathcal{P}'| = j+1$ exists, we can partition the midpoints $\mathcal{M}_{\mathcal{P}'}$ in a part which lies within $V(\mathcal{P})$ and one which lies outside. We call them $\mathcal{M}^i_{\mathcal{P}'} := \mathcal{M}_{\mathcal{P}'} \cap V(\mathcal{P})$ and $\mathcal{M}^o_{\mathcal{P}'} := \mathcal{M}_{\mathcal{P}'} \cap O$, respectively with $O := V(\mathcal{P}') \setminus V(\mathcal{P})$. Theorem 3 yields $|O| \leq 0.5j+3$ and thus $|\mathcal{M}^o_{\mathcal{P}'}| \leq 0.5j+3$. Basically, we can find an integer ℓ with $0 \leq \ell \leq 0.5j+3$ such that $|\mathcal{M}^i_{\mathcal{P}'}| = (j+1) - \ell$ and $|\mathcal{M}^o_{\mathcal{P}'}| = \ell$. In step 10 we run through every such ℓ until we reach $0.3251j$. For any choice of ℓ, in step 11 we cycle through all possibilities of choosing sets $S_i \subseteq V(\mathcal{P})$ and $S_o \subseteq V \setminus V(\mathcal{P})$ such that $|S_i| = (j+1) - \ell$ and $|S_o| = \ell$. Here S_i and S_o are candidates for $\mathcal{M}^i_{\mathcal{P}'}$ and $\mathcal{M}^o_{\mathcal{P}'}$, respectively. For any choice of S_i and S_o we try to construct a P_2-packing. If we succeed once we can return the desired larger P_2-packing. Otherwise we reach the point where $\ell = 0.3251j$. At this point we change our strategy. Instead of looking for the midpoints of $\mathcal{P}'$ we focus on the endpoints. We do so because this will improve the run time as we will see later. O is the disjoint union of $\mathcal{M}^o_{\mathcal{P}'}$ and the endpoints of $\mathcal{P}'$ which do not lie in $V(\mathcal{P})$ which we call $E^o_{\mathcal{P}'}$. At this point we must have $|\mathcal{M}^o_{\mathcal{P}'}| > 0.3251j$ and therefore $|E^o_{\mathcal{P}'}| < 0.1749j + 3$ as $O \leq 0.5j + 3$. Now there must be an integer $\bar{\ell}$ with $0 \leq \bar{\ell} \leq 0.1785j + 3$ such that $|E^o_{\mathcal{P}'}| = \bar{\ell}$ and the number of endpoints within $V(\mathcal{P})$ (called $E^i_{\mathcal{P}'}$) must be $2(j+1) - \bar{\ell}$. In step 13 we iterate through $\bar{\ell}$. In the next step we cycle through all candidate sets for $E^o_{\mathcal{P}}$ and $E^i_{\mathcal{P}}$ which are called B_i and B_o in the algorithm.

In step 15 we consider all possibilities $(e_1^1, e_2^1), \ldots, (e_1^{j+1}, e_2^{j+1})$ of how to pair the vertices in $B_i \cup B_o$. A pair of endpoints (e_r^s, e_{r+1}^s) means that both vertices should appear in the same P_2 of $\mathcal{P}'$. Finally, we try to construct $\mathcal{P}'$ from $(e_1^1, e_2^1), \ldots, (e_1^{j+1}, e_2^{j+1})$ by computing a matching according to [10].

3.3 Running Time

The only exponential run time contribution comes from the **for**-loops in Alg. 1. For any ℓ we execute step 10 at most $\binom{3j}{(j+1)-\ell}\binom{4j}{\ell} \in \mathcal{O}\left(\binom{3j}{j-\ell}\binom{4j}{\ell}\right)$ times, since $|V(\mathcal{P})| = 3j$ and $|V \setminus V(\mathcal{P})| \leq 4j$ due to Theorem 1. Likewise, $\mathcal{O}\left(\binom{3j}{2j-\ell}\binom{4j}{\ell}\right)$ upperbounds step 13.

Lemma 5. *For any integer z with $0 \leq z \leq 0.5j - 1$ the following holds:*
1. $\binom{3j}{j-z}\binom{4j}{z} < \binom{3j}{j-(z+1)}\binom{4j}{z+1}$; *and* 2. $\binom{3j}{2j-z}\binom{4j}{z} < \binom{3j}{2j-(z+1)}\binom{4j}{z+1}$.

Proof. 1. We have $\binom{3j}{j-(z+1)}\binom{4j}{z+1} - \binom{3j}{j-z}\binom{4j}{z} = \frac{(3j)!(4j)!((j-z)(4j-z)-(2j+z+1)(z+1))}{(j-z)!(2j+z+1)!(z+1)!(4j-z)!}$.
Now it is enough to show $(j-z)(4j-z) - (2j+z+1)(z+1)) > 0$ which evaluates to $4j^2 - 7jz - 2j - 2z - 1 > 0$. For the given z this always is true.

2. We have $\binom{3j}{2j-(z+1)}\binom{4j}{z+1} - \binom{3j}{2j-z}\binom{4j}{z} = \frac{(3j)!(4j)!((2j-z)(4j-z)-(j+z+1)(z+1))}{(2j-z)!(j+z+1)!(z+1)!(4j-z)!}$.
Then $((2j-z)(4j-z) - (j+z+1)(z+1)) = 8j^2 - 7jz - j - 2z - 1$ which for the given z is greater than zero. □

With Lemma 5 step 10 is upperbounded by $\mathcal{O}\left(\binom{3j}{(0.6749)j}\binom{4j}{0.3251j}\right)$ and step 13 by $\mathcal{O}\left(\binom{3j}{1.8251j}\binom{4j}{0.1749j}\right)$. Both are dominated by $\mathcal{O}(15.285^j)$. Notice the asymptotic speed-up we achieve by changing the strategy (WIN-WIN).

Theorem 4. P_2-PACKING *can be solved in time* $\mathcal{O}^*(2.482^{3k})$.

4 Future Work

It would be nice to derive smaller kernels than $7k$ or $1.5k$ for P_2-PACKING or TOTAL EDGE COVER, resp., in view of the mentioned lower bound results [2].

A closely related problem is MAXIMUM P_3-PACKING for which R. Hassin and S. Rubinstein [7] found a $\frac{3}{4}$-approximation. We try to apply extremal combinatorial methods to save colors for P_d-packings for $d \geq 3$. First results seem to be promising. So, a detailed combinatorial (extremal structure) study of (say graph) structure under the perspective of a specific combinatorial problem seems to pay off not only for kernelization (see [5]), but also for iterative approaches.

Developing exact algorithms for MAXIMUM P_2-PACKING would be interesting. Dynamic programming yields an $\mathcal{O}^*(2^n)$-algorithm. By Theorem 2, TOTAL EDGE COVER can be solved in time $\mathcal{O}^*(2^{1.5k}) \subseteq \mathcal{O}^*(2.829^k)$. Improving on exact algorithmics would also improve on the parameterized algorithm for TOTAL EDGE COVER. Alternatively, find a search-tree algorithm for TOTAL EDGE COVER.

We finally mention that H. Fernau, J. Kneis and P. Rossmanith could show that also the general TEST COVER problem is in $\mathcal{FPT}$, a bit surprising in view of the fact that the quite similar FEATURE SET problem is W[2]-complete [4]. However, the general algorithm is far from practical and needs to be improved.

References

1. K. M. J. De Bontridder, B. V. Halldórsson, M. M. Halldórsson, J. K. Lenstra, R. Ravi, and L. Stougie. Approximation algorithms for the test cover problem. *Math. Progr., Ser. B*, 98:477–491, 2003.
2. J. Chen, H. Fernau, Y. A. Kanj, and G. Xia. Parametric duality and kernelization: lower bounds and upper bounds on kernel size. *SIAM Journal on Computing*, 37:1077–1108, 2007.
3. J. Chen, S. Lu, S.-H. Sze, and F. Zhang. Improved algorithms for path, matching, and packing problems. In N. Bansal, K. Pruhs, and C. Stein, editors, *Symposium on Discrete Algorithms SODA*, pages 298–307. SIAM, 2007.
4. C. Cotta and P. Moscato. On the parameterized complextiy of problems related with feature identification for gene expression data mining techniques. *Bioinformatics*, 1:1–8, 2002.
5. V. Estivill-Castro, M. R. Fellows, M. A. Langston, and F. A. Rosamond. FPT is P-time extremal structure I. In H. Broersma, M. Johnson, and S. Szeider, editors, *Algorithms and Complexity in Durham ACiD 2005*, volume 4 of *Texts in Algorithmics*, pages 1–41. King's College Publications, 2005.
6. H. Fernau and D. F. Manlove. Vertex and edge covers with clustering properties: Complexity and algorithms. In *Algorithms and Complexity in Durham ACiD 2006*, pages 69–84. King's College, London, 2006.
7. R. Hassin and S. Rubinstein. An approximation algorithm for maximum of 3-edge paths. *Information Processing Letters*, 63:63–67, 1997.
8. R. Hassin and S. Rubinstein. An approximation algorithm for maximum triangle packing. *Discrete Applied Mathematics*, 154:971–979; 2620 [Erratum], 2006.
9. P. Hell and D. G. Kirkpatrick. Star factors and star packings. Technical Report 82-6, Computing Science, Simon Fraser University, Burnaby, B.C. V5A1S6, Canada, 1982.
10. W. Jia, C. Zhang, and J. Chen. An efficient parameterized algorithm for m-set packing. *Journal of Algorithms*, 50:106–117, 2004.
11. D. G. Kirkpatrick and P. Hell. On the completeness of a generalized matching problem. In *ACM Symposium on Theory of Computing STOC*, pages 240–245, 1978.
12. Y. Liu, S. Lu, J. Chen, and S.-H. Sze. Greedy localization and color-coding: improved matching and packing algorithms. In H. L. Bodlaender and M. Langston, editors, *International Workshop on Parameterized and Exact Computation IWPEC*, volume 4169 of *LNCS*, pages 84–95. Springer, 2006.
13. E. Prieto and C. Sloper. Looking at the stars. In R. Downey, M. Fellows, and F. Dehne, editors, *International Workshop on Parameterized and Exact Computation IWPEC 2004*, volume 3162 of *LNCS*, pages 138–148. Springer, 2004.
14. J. Wang and Q. Feng. An $O^*(3.52^{3k})$ parameterized algorithm for 3-set packing. In M. Agrawal et al., editors, *Theory and Applications of Models of Computation TAMC*, volume 4978 of *LNCS*, pages 82–93. Springer, 2008.
15. J. Wang, D. Ning, Q. Feng, and J. Chen. An improved parameterized algorithm for a generalized matching problem. In M. Agrawal et al., editors, *Theory and Applications of Models of Computation TAMC*, volume 4978 of *LNCS*, pages 212–222. Springer, 2008.