Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 18, no. 4, pp. 515–538 (2014)
DOI: 10.7155/jgaa.00335

Parameterized Algorithms for the H-Packing with t-Overlap Problem

Alejandro López-Ortiz Jazmín Romero

David R. Cheriton School of Computer Science,
University of Waterloo, Waterloo, ON. Canada

Abstract

We introduce the k-H-Packing with t-Overlap problem to formalize the problem of discovering overlapping communities in real networks. More precisely, in the k-H-Packing with t-Overlap problem, we search in a graph G for at least k subgraphs each isomorphic to a graph H such that any pair of subgraphs shares at most t vertices. In contrast with previous work where communities are disjoint, we regulate the overlap through a variable t. Our focus is on the parameterized complexity of the k-H-Packing with t-Overlap problem.

Here, we provide a new technique for this problem generalizing the crown decomposition technique [2]. Using our global rule, we achieve a kernel with size bounded by $2(rk - r)$ for the k-K_r-Packing with $(r-2)$-Overlap problem. That is, when H is a clique of size r and $t = r - 2$.

In addition, we introduce the first parameterized algorithm for the k-H-Packing with t-Overlap problem when H is an arbitrary graph of size r. Our algorithm combines a bounded search tree with a greedy localization technique and runs in time $O(r^{rk}k^{(r-t-1)k+2}n^r)$, where $n = |V(G)|$, $r = |V(H)|$, and $t < r$.

Finally, we apply this search tree algorithm to the kernel obtained for the k-K_r-Packing with $(r-2)$-Overlap problem, and we show that this approach is faster than applying a brute-force algorithm in the kernel. In all our results, r and t are constants.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
March 2014	June 2014	July 2014	July 2014	December 2014

Published: December 2014

Article type: Regular paper — Communicated by: S. Pal and K. Sadakane

Part of these results appeared in WALCOM 2014 [17].

E-mail addresses: alopez-o@uwaterloo.ca (Alejandro López-Ortiz) hjromero@uwaterloo.ca (Jazmín Romero)

1 Introduction

Many complex systems that exist in real applications can be represented by networks, where each node is an entity, and each edge represents a relationship [12]. A community is a part of the network in which the nodes are more highly interconnected to each other than to the rest [15]. To extract these communities is known as the *community discovering problem* [1]. There are approaches for this problem that determine separate communities [1, 8]. However, most real networks are characterized by well-defined communities that share members with others [4, 15]. Moreover, these approaches model a community as a fixed graph H, when in real applications there are different models for communities. To overcome these deficiencies, we introduce the *k-$\mathcal{H}$-Packing with t-Overlap problem* as a more realistic formalization of the community discovering problem. To the best of our knowledge, the k-$\mathcal{H}$-Packing with t-Overlap problem has not been studied before.

In the k-$\mathcal{H}$-Packing with t-Overlap problem, the goal is to find at least k subgraphs (the communities) in a graph G (the network) such that each subgraph is isomorphic to a member of a family $\mathcal{H}$ of graphs (the community models) and each pair of subgraphs can overlap in at most t vertices (the shared members)[1].The formal definition of this problem is as follows.

The k-$\mathcal{H}$-Packing with t-Overlap problem
Input: A family $\mathcal{H}$ of graphs, a graph G, and non-negative integers k and t.
Question: Does G contain at least k subgraphs $\mathcal{K} = \{S_1^*, \ldots, S_k^*\}$ where each S_i^* is isomorphic to a member of $\mathcal{H}$ and every pair S_i^*, S_j^* can overlap in at most t vertices, $|V(S_i^*) \cap V(S_j^*)| \leq t$, for $i \neq j$?

When the family $\mathcal{H}$ is composed of only one graph H, the problem is simply denoted as the k-H-Packing with t-Overlap problem.

The k-$\mathcal{H}$-Packing with t-Overlap problem is a generalization of the well-studied *k-$\mathcal{H}$-Packing problem* which seeks for vertex-disjoint subgraphs. The k-$\mathcal{H}$-Packing with t-Overlap problem is NP-complete. This follows since every instance of the k-$\mathcal{H}$-packing problem, which is NP-complete [10], is mapped to an instance of the k-$\mathcal{H}$-Packing with t-Overlap problem by making $t = 0$. In this work, our goal is to design *fixed-parameter algorithms* or *FPT-algorithms*; that is, algorithms with running time polynomial in the input size n but exponential in a specified parameter k, i.e., $f(k)n^{O(1)}$. We also seek for finding *problem kernels*; that is, reduced instances with size bounded by $f(k)$.

Related Results. As mentioned before, the k-H-Packing with t-Overlap problem generalizes the problem of packing vertex-disjoint subgraphs. Parameterized results for this problem have been obtained in [6, 7, 16]. The latest result is a kernel of size $O(k^{|V(H)|-1})$ for packing an arbitrary graph H by Moser [14].

[1] To follow standard notation with packing and isomorphism problems, the meaning of G and $\mathcal{H}$ have been exchanged with respect to their meaning in [17].

Another related problem to the k-H-packing with t-Overlap, when H is a clique, is the *cluster editing problem*. This problem consists of modifying a graph G by adding or deleting edges such that the modified graph is composed of a vertex-disjoint union of cliques. Some works have considered overlap [3, 5]. Fellows et al. [5] allow that each vertex of the modified graph can be contained in at most s maximal cliques.

The problem of finding one community of size at least r in a given network is also related to the k-H-Packing with t-Overlap problem. The most studied community models are cliques and some relaxations of cliques. Parameterized complexity results for this problem can be found in [9, 11, 19]. Overlap has not yet been considered under that setting.

Our Results. Besides introducing the k-$\mathcal{H}$-Packing with t-Overlap problem, we provide here a study of its parameterized complexity. First, we introduce a global reduction rule, the *clique-crown reduction rule*, based on a non-trivial generalization of the crown decomposition technique [2]. To the best of our knowledge, the crown decomposition technique has not been adapted to obtain kernels for problems that find subgraphs with arbitrary overlap. Using our clique-crown decomposition rule, we achieve a problem kernel of size $2(rk - r)$ for the k-H-Packing with t-Overlap problem when H is a clique with r vertices, i.e., a K_r, and $t = r - 2$.

We also provide an $O(r^{rk}k^{(r-t-1)k+2}n^r)$ running time algorithm for the k-H-packing with t-Overlap for any arbitrary graph H of size r and any overlap value $t < r$. Our algorithm is a non-trivial generalization of the search tree algorithm to find disjoint triangles presented by Fellows et al. [6]. We use a bounded search tree together with a greedy localization technique. The analysis of our algorithm is novel since we allow overlap between subgraphs. Even though the k-H-packing problem (the vertex-disjoint version) is well studied, our search tree algorithm is the first one to consider variable overlap between subgraphs.

In addition, we apply our search tree algorithm to the $2(rk - r)$ kernel of the k-K_r-Packing with $(r - 2)$-Overlap problem, obtaining in this way an $O(r^{3k+r-1}k^{2k+3r} + n^r)$ running time algorithm. This approach is faster than solving the k-K_r-Packing with $(r - 2)$-Overlap problem by brute-force on the kernel. In all our results, r and t are constants.

This paper is organized as follows. In Section 2, we introduce the terminology and notation used in the paper. Section 3 describes the reduction rules as well as our kernelization algorithm for the k-K_r-Packing with t-Overlap problem. In Section 4, we describe the details of our search tree algorithm for the k-H-Packing with t-Overlap problem. Section 5 provides an FPT-algorithm for the k-K_r-Packing with $(r - 2)$-Overlap problem. Finally, Section 6 states the conclusion of this work.

2 Terminology and Notation

All graphs in this document are undirected, simple, and connected. For a graph G, $V(G)$ and $E(G)$ denote its sets of vertices and edges, respectively. Two subgraphs S and P are vertex-disjoint if they do not share vertices, i.e., $V(S) \cap V(P) = \emptyset$. Otherwise, we say that S and P *overlap* in $|V(S) \cap V(P)|$ vertices. We extend this terminology when S and P are sets of vertices instead. For a set of vertices $A \subseteq V(G)$, the neighborhood of A is defined as $N(A) = \{v \notin A \mid (u,v) \in E(G) \text{ and } u \in A\}$. The subgraph induced by A in G is denoted as $G[A]$. For a set of subgraphs C, $|C|$ is the number of subgraphs in C while $V(C) = \bigcup_{i=1}^{|C|} V(C_i)$ where $C_i \in C$.

A subgraph isomorphic to H will be called an *H-subgraph*. In the specific case where H is a clique of r vertices, i.e., a K_r, an H-subgraph will be referred to as an *r-clique*.

An *l-H-Packing with t-Overlap* $\mathcal{L}$ is a set of l H-subgraphs $\mathcal{L} = \{Q_1, \ldots, Q_l\}$ of G where every pair Q_i, Q_j overlaps in at most t vertices. If $l \geq k$ then $\mathcal{L}$ will be called a *k-solution* and will be represented by the letter $\mathcal{K}$. An H-Packing with t-Overlap $\mathcal{M}$ is *maximal* if any H-subgraph of G that is not already in $\mathcal{M}$ overlaps in more than t vertices with some H-subgraph in $\mathcal{M}$.

For any pair of disjoint sets of vertices A and B such that $|A| + |B| = r$, we say that A is a *sponsor* of B (or vice versa) if $G[A \cup B]$ is an H-subgraph. $\mathsf{Sponsors}(A)$ is the set of sponsors of A in $V(G)$. We use the term *complete* A to represent the selection of a sponsor B in $\mathsf{Sponsors}(A)$ to update A as $A \cup B$. The resulting H-subgraph $G[A \cup B]$ is called an *H-completed subgraph*, and it is denoted as $A \cdot B$. Figure 1 shows an instance of the k-K_5-Packing with 1-Overlap ($H = K_5$ and $t = 1$). In this instance, the sets of vertices $\{1,4,5\}$ and $\{13,14\}$ form a K_5 $G[\{1,4,5,13,14\}]$; thus, the set $\{13,14\}$ is a sponsor of the set $\{1,4,5\}$. Other sponsors of $\{1,4,5\}$ are $\{15,16\}$ and $\{2,3\}$.

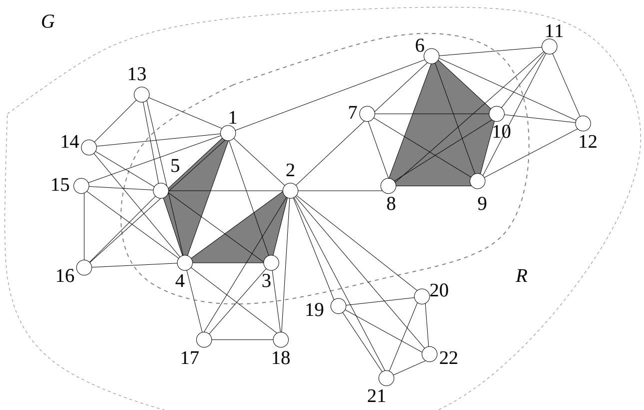

Figure 1: An instance of the k-K_5-Packing with 1-Overlap problem ($H = K_5$ and $t = 1$).

The notation is also applicable when A and B are a pair of vertex-disjoint cliques instead of sets of vertices. In this case, $G[V(A) \cup V(B)] = A \cdot B$ is an r-clique. It is worth to emphasize that, even if $G[V(A) \cup V(B)]$ is a clique but does not have r vertices then B does not complete A.

3 Reduction Rules for the k-K_r-Packing with t-Overlap Problem

In this section, we present our clique-crown reduction rule for the k-K_r-Packing with t-Overlap problem which is based on our clique-crown decomposition. We also introduce a method to compute such decomposition. Finally, we present an algorithm that reduces the k-K_r-Packing with t-Overlap problem to a kernel when $t = r - 2$.

3.1 Preliminaries

A parameterized problem is reduced to a *problem kernel*, if any instance can be reduced to a smaller instance such that: the reduction is in polynomial time, the size of the new instance is depending only on an input parameter, and the smaller instance has a solution if and only if the original instance has one.

Our goal is to reduce the k-K_r-Packing with t-Overlap problem to a problem kernel. The formal definition of our studied problem is as follows. Let $0 \leq t \leq r - 1$ be fixed in the following definition.

k-K_r-Packing with t-Overlap problem
Instance: A graph G and a non-negative integers k.
Parameter: k
Question: Does G contain a set of r-cliques $\mathcal{K} = \{S_1, \ldots, S_l\}$ for $l \geq k$, such that $|V(S_i) \cap V(S_j)| \leq t$, for any pair S_i, S_j and $i \neq j$?

Our clique-crown decomposition is a generalization of the crown decomposition technique. This technique was introduced by Chor et al. [2], and it has been adapted to obtain kernels for packing problems [6, 13, 16].

Definition 1 *A* crown decomposition (H, C, R) *in a graph* G *is a partitioning of* $V(G)$ *into three sets* H, C, *and* R *that have the following properties:*

1. $C = C_m \cup C_u$ *(the crown) is an independent set in* G.
2. H *(the head) is a separator in* G *such that there are no edges in* G *between vertices belonging to* C *and vertices belonging to* R.
3. R *is the rest of the graph, i.e.,* $R = V(G) \backslash (C \cup H)$.
4. *There is a perfect matching between* C_m *and* H.

Generally, vertices in C_m and in H are part of a desired solution while vertices in C_u can be removed from G.

A crown decomposition can be computed in polynomial time for a graph G given certain conditions.

Lemma 1 *[2] If a graph $G = (V, E)$ has an independent set of vertices $I \subseteq V(G)$ such that $|I| \geq |N(I)|$, then G has a crown decomposition, where $C \subseteq I$ and $H \subseteq N(I)$, that can be found in time $O(|V(G)| + |E(G)|)$, given I.*

We next apply a natural reduction rule to the input graph G.

Reduction Rule 1 *Delete any vertex v and any edge e that are not included in a K_r of G.*

To further reduce the graph G, we design the clique-crown reduction rule which is based on our proposed clique-crown decomposition.

3.2 The Clique-Crown Reduction Rule

In the clique-crown decomposition, we have cliques in both the head H and the crown C, and each clique in H is completed by at least one clique in C.

Definition 2 *A* clique-crown decomposition *(H, C, R) is a partition of G that have the following properties:*

1. *$C = C_m \cup C_u$ (the crown) is a set of cliques in G where each clique has size at most $r - (t + 1)$. Cliques in C are denoted with letters $\alpha, \beta, \ldots$.*
2. *H (the head) is a set of cliques in G where each clique has size at least $t + 1$ and at most $r - 1$. Cliques in H are denoted with letters $\mathbb{A}, \mathbb{B}, \ldots$. The head satisfies the following conditions.*
   - i. *Each $\mathbb{A} \in H$ is completed by at least one clique in C. Furthermore, none subgraph $\mathbb{A}'$ of $\mathbb{A}$ is completed by a clique in C.*
   - ii. *Each pair of cliques $\mathbb{A}$ and $\mathbb{B}$ in H overlaps in at most t vertices.*
   - iii. *Each $\mathbb{A} \in H$ is completed by a clique in R, defined below. In addition, the size of any subgraph $\mathbb{A}'$ of $\mathbb{A}$ that is completed by a clique in R is at most t.*
3. *R is the rest of the graph, i.e., $R = G[V(G) \backslash (V(C) \cup V(H))]$.*
4. *The set of vertices of the cliques in H, $V(H)$, is a separator such that there are no edges in G from C to R.*
5. *There exists an injective function f mapping each clique $\mathbb{A} \in H$ to a distinct clique $\alpha \in C_m$ such that α completes $\mathbb{A}$. In this way, $\mathbb{A} \cdot \alpha$ is an r-clique that we call* a mapped r-clique. *We impose the condition that any pair of mapped r-cliques overlaps in at most t vertices.*

Figure 2 shows an example of a clique-crown decomposition for an instance of the 5-K_4-Packing with 1-Overlap problem ($k = 5$, $r = 4$, and $t = 1$). Cliques that belong to the head H are $G[\{3, 4\}]$, $G[\{8, 9, 10\}]$, and $G[\{11, 12\}]$. These cliques are highlighted with thicker lines. Cliques in the clique-crown C are

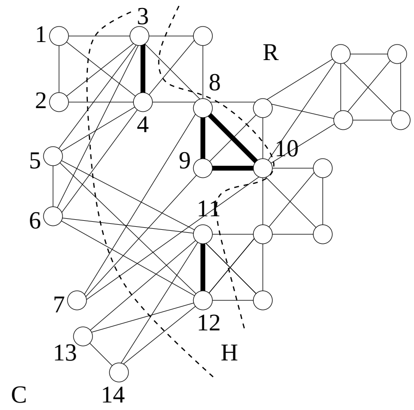

Figure 2: Example of a clique-crown decomposition for an instance of the 5-K_4-Packing with 1-Overlap problem ($k = 5$, $r = 4$, and $t = 1$).

$G[\{1,2\}]$, $G[\{5,6\}]$, $G[\{7\}]$ and $G[\{13,14\}]$. The mapped-cliques in this example are $G[\{1,2,3,4\}]$, $G[\{7,8,9,10\}]$, and $G[\{11,12,13,14\}]$. Note that the clique $G[\{5,6\}]$ is in C_u.

To design the clique-crown reduction rule, we use an annotated version of the k-K_r-Packing with t-Overlap problem. In this annotated version, any r-clique of the solution overlaps in at most t vertices with any clique from a set $\mathcal{F}$ given as part of the input. Again, let t be fixed in the following definition.

Definition 3 ***Annotated K_r-Packing with t-Overlap problem***
Instance: *A graph G, a set of cliques $\mathcal{F}$ from G where any clique in $\mathcal{F}$ has size at least $t+1$ and at most $r-1$, and a non-negative integer k.*
Parameter: $k - |\mathcal{F}|$
Question: *Does G contain a set of r-cliques $\mathcal{K} = \{S_1, S_2, ..., S_l\}$ for $l \geq k - |\mathcal{F}|$, such that $|V(S_i) \cap V(S_j)| \leq t$, for any pair S_i, S_j ($i \neq j$), and $|V(S) \cap V(C)| \leq t$ for any $S \in \mathcal{K}$ and $C \in \mathcal{F}$?*

Reduction Rule 2 The Clique-Crown Reduction. *If G admits a clique-crown decomposition (H, C, R) then reduce G as $G' = G[V(G) \backslash V(C)]$ and $k = k - |H|$. Make H be the set of cliques $\mathcal{F}$ of the annotated K_r-Packing with t-Overlap problem.*

The goal of the clique-crown reduction is to make the mapped r-cliques part of the solution and remove unnecessary vertices from G. As part of the correctness of Rule 2, we prove first that the vertices in $V(C_u) \backslash V(C_m)$ are not included in any r-clique of the solution.

Lemma 2 *The instance (G, k) has a k-K_r-Packing with t-Overlap if and only if the instance $(G \backslash (V(C_u) \backslash V(C_m)), k)$ has a k-K_r-Packing with t-Overlap.*

Proof: Cliques in C_u only complete cliques from the set H; otherwise $V(H)$ would not be a separator. However, every clique in H is mapped to a clique in C_m by the injective function. On the other hand, by Definition 2, cliques in H cannot be partitioned in more cliques than $|H|$ that could be completed by cliques in C_u.

□

We use the next observation for the proof of correctness of the following lemmas.

Observation 1 *Any clique $\mathbb{A}'$ of $\mathbb{A} \in H$, where $|V(\mathbb{A}')| \geq t+1$, is an induced subgraph of at most one r-clique of any solution, since the k-K_r-Packing with t-Overlap problem allows overlap at most t.*

Lemma 3 *If G admits a clique-crown decomposition (H, C, R), then the set of mapped r-cliques is an $|H|$-K_r-Packing with t-Overlap in G.*

Proof: Follows from Definition 2. □

The input graph G' for the annotated K_r-packing with t-Overlap problem is obtained by removing C from G. Since a clique $\mathbb{A} \in H$ is already an induced subgraph of an r-clique from the solution, $\mathbb{A}$ cannot be an induced subgraph of another r-clique from the solution (Observation 1). Hence, we make the set H to be the set of cliques $\mathcal{F}$ in the annotated K_r-packing with t-Overlap problem. In the example of Figure 2, $\mathcal{F}$ would have $G[\{3,4\}]$, $G[\{8,9,10\}]$ and $G[\{11,12\}]$.

Lemma 4 *Let $G' = G[V(G) \backslash V(C_m \cup C_u)]$. The instance (G, k) has a k-K_r-Packing with t-Overlap if and only if the instance $(G', H, k - |H|)$ has an annotated $(k - |H|)$-K_r-packing with t-Overlap.*

Proof: Assume by contradiction that G admits a clique-crown decomposition (H, C, R) and has a k-solution, but $(G', H, k - |H|)$ does not have an annotated solution. By Lemma 2, vertices in $V(C_u) \backslash V(C_m)$ are redundant. That is, they do not belong to any r-clique of the solution. By Observation 1, every $\mathbb{A} \in H$ is in at most one r-clique of the solution. Therefore, we cannot form more than $|H|$ r-cliques by completing each clique of H with cliques in R rather than with cliques in C_m. The only case that we could have more than $|H|$ r-cliques is if there is a clique $\mathbb{A} \in H$ that has at least two cliques $\mathbb{A}', \mathbb{A}''$ each of size at least $t+1$ that are completed by some clique in R. However, that is not possible by Definition 2.

Assume now that $(G', H, k - |H|)$ has an annotated $(k - |H|)$-K_r-Packing with t-Overlap, but (G, k) does not have a k-K_r-Packing with t-Overlap. This would imply that the sets H and C form more than $|H|$ r-cliques which is a contradiction by Lemma 3. □

The next claim states how a solution of the original instance can be obtained using an annotated solution of the reduced graph G'.

Claim 1 *Let $\mathcal{K}'$ be an annotated $(k-|H|)$-K_r-Packing with t-Overlap of G' and $\mathcal{H}$ be the set of mapped r-cliques in G found with the clique-crown decomposition. The set $\mathcal{K}' \cup \mathcal{H}$ is a k-K_r-Packing with t-Overlap of G.*

Proof: Since, $G' = G[V(G)\backslash V(C)]$, $\mathcal{K}'$ is a $(k-|H|)$-K_r-Packing with t-Overlap of G as well. By Lemma 3, the set $\mathcal{H}$ is an $|H|$-K_r-Packing with t-Overlap in G. Since H becomes the set of cliques $\mathcal{F}$ in the annotated instance, then no r-clique of $\mathcal{K}'$ overlaps in more than t vertices with any r-clique of $\mathcal{H}$. Therefore, $\mathcal{K}' \cup \mathcal{H}$ is a k-K_r-Packing with t-Overlap in G. □

In Figure 3, the set of mapped cliques of Figure 2 are highlighted with thicker lines. An annotated solution for the reduced graph is indicated with dashed lines. We can see how the set of mapped cliques and the annotated solution form a 5-K_4-Packing with 1-Overlap.

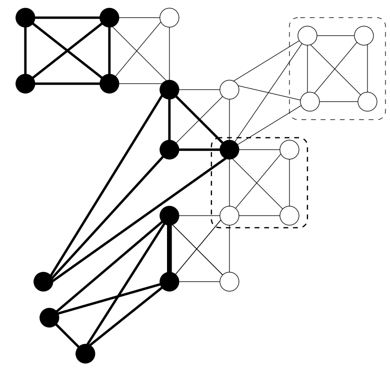

Figure 3: An annotated solution for an instance of the 5-K_4-Packing with 1-Overlap problem is indicated with dashed lines ($k = 5$, $r = 4$, $t = 1$, and $|H| = 3$).

Computing the Clique-Crown Decomposition

We next present a method to find a clique-crown decomposition in G given two sets of cliques: O and $\mathsf{Cliques}(O)$. These sets of cliques follow the next conditions. Each clique in O has size at most $r-(t+1)$, any pair of cliques in $\mathsf{Cliques}(O)$ overlaps in at most t vertices, and every clique $\mathbb{A} \in \mathsf{Cliques}(O)$ should be completed by at least one clique in O. In addition, none subgraph $\mathbb{A}'$ of $\mathbb{A}$ is completed by a clique in $\mathsf{Cliques}(O)$, and the size of any subgraph $\mathbb{A}'$ of $\mathbb{A}$ that is completed by a clique in $G[V(G)\backslash(V(O) \cup V(\mathsf{Cliques}(O)))]$ is at most t. Observe that the size of $\mathbb{A}$ is at least $t+1$ and at most $r-1$.

The following method is a generalization of the method used to compute a crown-decomposition for the edge disjoint K_3-packing problem [16].

Lemma 5 *Any graph G with a set O of vertex-disjoint cliques where each clique has size at most $r-(t+1)$ and $|O| \geq |\mathsf{Cliques}(O)|$, has a clique-crown decomposition (H, C, R) where $H \subseteq \mathsf{Cliques}(O)$, that can be found in $O(|V(G)|+|E(G)|)$ time given O and $\mathsf{Cliques}(O)$.*

Proof: First, we construct a graph G' from G as follows. We initialize $V(G') = V(G)$ and $E(G') = E(G)$. We contract in G' each clique $\alpha \in O$ into a single vertex v_α, and we denote the set of contracted cliques as O_{cont}. After that for each clique $\mathbb{A} \in \mathsf{Cliques}(O)$, we add a vertex $v_\mathbb{A}$ to $V(G')$, i.e., *a representative vertex*; we denote as *Rep* the set of all representative vertices. We say that v_α "completes" $v_\mathbb{A}$ if the clique α completes the clique $\mathbb{A}$. For every vertex $v_\alpha \in O_{cont}$ that completes $v_\mathbb{A}$, add $(v_\alpha, v_\mathbb{A})$ to $E(G')$. After that, add to $E(G')$ an edge from $v_\mathbb{A}$ to each vertex of $\mathbb{A}$. Finally, remove from $E(G')$ the edges from v_α to $\mathbb{A}$.

We next show that G' has a crown decomposition (H', C', R'). In G', the set of contracted cliques O_{cont} is an independent set. By the construction of G', we know that $N(O_{cont})$ is the set of representative vertices *Rep*. Since we introduced a representative vertex per clique in $\mathsf{Cliques}(O)$, then $|N(O_{cont})| = |\mathsf{Cliques}(O)|$. Thus, since $|O| \geq |\mathsf{Cliques}(O)|$ then $|O_{cont}| \geq |N(O_{cont})|$ in G'. By Lemma 1, G' admits a crown decomposition (H', C', R') that is computed in polynomial time, where $C' \subseteq O_{cont}$ and $H' \subseteq N(O_{cont}) = Rep$.

Now, we use the crown decomposition (H', C', R') of G' to construct the clique-crown decomposition (H, C, R) of G where $H \subseteq \mathsf{Cliques}(O)$.

1. For each vertex $v_\alpha \in C' \subseteq O_{cont}$, add the clique α to C. The size of each clique in C is at most $r-(t+1)$. This follows because $\alpha \in O$ and each clique in O has size at most $r-(t+1)$.

2. For each vertex $v_\mathbb{A} \in H'$, where $H' \subseteq Rep$, we assign to H the clique that this vertex represents, i.e., $\mathbb{A}$. Since $H \subseteq \mathsf{Cliques}(O)$, then each clique in H has size at least $t+1$ and at most $r-1$. Likewise, properties i-iii from Definition 2 follow.

3. $R = G[V(G) \backslash (V(C) \cup V(H))]$.

4. The set of vertices of the cliques in H, $V(H)$, is a separator. This follows since cliques in C complete only cliques on H; thus, vertices in $V(C)$ are only adjacent to vertices in $V(H)$.

5. We make the perfect matching between C' and H' correspond to the injective function f in the following way. For any matched edge $(v_\alpha, v_\mathbb{A})$ complete $\mathbb{A}$ with α. For any pair of mapped r-cliques $\mathbb{A} \cdot \alpha$ and $\mathbb{B} \cdot \beta$ completed in this way, $|V(\mathbb{A} \cdot \alpha) \cap V(\mathbb{B} \cdot \beta)| \leq t$. This follows because α and β are vertex-disjoint, and $|V(\mathbb{A}) \cap V(\mathbb{B})| \leq t$ by assumption in the set $\mathsf{Cliques}(O)$.

Thus, if $|O| \geq |\mathsf{Cliques}(O)|$ then G admits a clique-crown decomposition. $\square$

One method to obtain the sets O and $\mathsf{Cliques}(O)$ is to compute a maximal K_r-packing with t-Overlap $\mathcal{M}$ from G. O will be the set of all cliques in $G[V(G)\backslash V(\mathcal{M})]$. After applying the Reduction Rule 1, each clique in O completes at least one clique in $G[V(\mathcal{M})]$. $\mathsf{Cliques}(O)$ is therefore the set of cliques in $G[V(\mathcal{M})]$ completed by cliques in O. The overlap between an r-clique $\mathbb{A} \cdot \alpha$, where $\mathbb{A} \in \mathsf{Cliques}(O)$ and $\alpha \in O$, with some clique in $\mathcal{M}$ is at least $t+1$. Therefore, the size of $\mathbb{A}$ is at least $t+1$ and the size of α is at most $r-(t+1)$. It has to be verified if the sets O and $\mathsf{Cliques}(O)$ follow the properties of the crown and the head, respectively, from Definition 2.

3.3 A Kernel for the k-K_r-Packing with $(r-2)$-Overlap Problem

Using the clique-crown reduction rule, we introduce an algorithm to obtain the kernel for the k-K_r-packing with $(r-2)$-Overlap problem. First, we compute a maximal solution for the k-K_r-packing with $(r-2)$-Overlap problem. Next, we show that the sets O and $\mathsf{Cliques}(O)$ are composed of vertices that are outside and inside, respectively, of the maximal solution. The steps of the algorithm are outlined in Algorithm 1.

Algorithm 1 k-K_r-packing with $(r-2)$-Overlap Algorithm

```
Input: A graph G = (V, E) and a non-negative integer k.
Reduce G by Reduction Rule 1.
Greedily, find a maximal K_r-Packing with (r − 2)-Overlap M in G.
if |M| ≥ k then
  Accept
else
  Let O be V(G)\V(M) and Cliques(O) be the set of cliques in M completed
  by vertices in O.
  if |O| ≥ |Cliques(O)| then
    Apply the clique-crown reduction rule in G (Rule 2).
  end if
end if
```

We next introduce a series of lemmas that characterize the sets O and $\mathsf{Cliques}(O)$ defined in Algorithm 1.

Claim 2 *$O = V(G)\backslash V(\mathcal{M})$ is an independent set.*

Proof: Assume by contradiction that there exists an edge (u,v) in $G[O]$. After applying Reduction Rule 1, each edge in the reduced graph is included in at least one r-clique; thus, (u,v) belongs to at least one r-clique S'. S' is not in $\mathcal{M}$; otherwise u,v would not be in O. S' is not in O; otherwise as S' would be disjoint from $\mathcal{M}$, and hence, S' could be added to $\mathcal{M}$, contradicting the maximality of $\mathcal{M}$. Thus, S' should overlap with at least one r-clique $S \in \mathcal{M}$, for $S \neq S'$.

Since u, v are both in O, the overlap with S is at most $r-2$, i.e., $|V(S) \cap V(S')| = r-2$, but in this case, S' could be added to $\mathcal{M}$ contradicting the maximality of $\mathcal{M}$. □

Claim 3 *Each K_{r-1} T completed by any $u \in O$ is contained in an r-clique $S \in \mathcal{M}$.*

Proof: $V(T) \cap V(O) = \emptyset$. Assume otherwise that there is a vertex $v \in V(T)$ contained in O. However, since $T \cdot u$ forms an r-clique this would imply that there is an edge (u, v) in O which is a contradiction since O is an independent set. Thus, $V(T) \subset V(\mathcal{M})$.

We claim that $|V(T) \cap V(S)| = r-1$ for some $S \in \mathcal{M}$, i.e., $V(T) \subset V(S)$. Suppose otherwise that $|V(T) \cap V(S)| < r-1$, for any $S \in \mathcal{M}$. Since $u \in O$, this would imply that $|V(T \cdot u) \cap V(S)| \leq r-2$ for every $S \in \mathcal{M}$, and $T \cdot u$ could be added to $\mathcal{M}$ as the k-K_r-packing with $(r-2)$-Overlap problem allows overlap at most $r-2$, contradicting the assumption of maximality of $\mathcal{M}$. □

Claim 4 *Each clique $\mathbb{A} \in \mathsf{Cliques}(O)$ is completed by at least one clique in O and none subgraph $\mathbb{A}'$ of $\mathbb{A}$ is completed by a clique in O. In addition, the size of any subgraph $\mathbb{A}'$ of $\mathbb{A}$ completed by a clique in $G[V(G) \setminus (V(O) \cup V(\mathsf{Cliques}(O)))]$ is at most t. Furthermore, any pair of cliques in $\mathsf{Cliques}(O)$ overlaps in at most t vertices.*

Proof: Assume by contradiction that there is a clique $\mathbb{A}' \subset \mathbb{A}$ of size $s < r-1$ completed by a clique in $G[V(O)]$. This would imply that there is a K_{r-s} in $G[V(O)]$, a contradiction since O is an independent set (Claim 2).

The second and third parts of the claim follows because the size of each clique in $\mathsf{Cliques}(O)$ is $t+1 = r-1$ (Claim 3) and therefore the size of the largest subgraph of $\mathbb{A}$ is at most $r-2 = t$. □

We can see that by Claims 2 and 4, the sets O and $\mathsf{Cliques}(O)$ are supersets of the head H and the clique-crown C, respectively. Thus, the method described in proof of Lemma 5 can be used to compute a clique-crown decomposition in G. Next, we prove that the size of the reduced instance is bounded by a function of the parameter k.

Claim 5 *The set of vertices O completes at most $rk - r$ K_{r-1}'s.*

Proof: By Claim 3, vertices in O only complete K_{r-1}'s contained in K_r's in $\mathcal{M}$. There are r K_{r-1}'s in a K_r and at most $k-1$ K_r's in $\mathcal{M}$; thus, there are at most $rk - r$ K_{r-1}'s that can be completed by vertices in O. □

Claim 6 $|O| < rk - r$

Proof: In Algorithm 1, if $|O| \geq |\mathsf{Cliques}(O)|$, O is reduced by the clique-crown reduction rule (Rule 2). Since $|\mathsf{Cliques}(O)| < rk - r$ then $|O| < rk - r$, after applying that rule. □

Lemma 6 *If* $|V(G)| > 2(rk - r)$ *then Algorithm 1 will either find a* k-K_r-*Packing with* t-*Overlap, or it will reduce* G.

Proof: Assume by contradiction that $|V(G)| > 2(rk - r)$, but the algorithm neither finds a k-K_r-packing with $(r-2)$-Overlap nor reduces the graph G. Any vertex $v \in V(G)$ that was not reduced by Rule 1 is in $V(\mathcal{M})$, or it is in $O = V(G) \backslash V(\mathcal{M})$; thus, $|V(G)| = |V(\mathcal{M})| + |O|$.

The size of $\mathcal{M}$ is at most $k-1$; thus, $|V(\mathcal{M})|$ is at most $rk-r$, and by Claim 6 we know that an upper bound for $|O|$ is $rk - r$.

In this way, the size of the instance is at most $2(rk - r)$ which contradicts the assumption that $|V(G)| > 2(rk - r)$. □

Claim 7 *The* k-K_r-*packing with* $(r-2)$-*Overlap problem admits a* $2(rk - r)$ *kernel which can be found in* $O(n^r)$ *time.*

Proof: By Lemma 6, the reduced instance has size at most $2(rk - r)$. Rule 1 is computed in time $O(n^r)$, which is also the same time to compute the maximal solution $\mathcal{M}$ and $\mathsf{Cliques}(O)$. Lemma 5 shows that the clique-crown decomposition is computed in polynomial time given the set of cliques O and $\mathsf{Cliques}(O)$. The set O corresponds to the independent set $V(G) \backslash V(\mathcal{M})$ (Lemma 2), and the set $\mathsf{Cliques}(O)$ is the set of K_{r-1}'s completed by vertices in O. By Claim 3, all these K_{r-1}'s are contained in the K_r's of $\mathcal{M}$. Moreover, in Rule 1, we already compute all K_{r-1}'s that a vertex completes. Thus, the time to obtain $\mathsf{Cliques}(O)$ is $O(n^r)$. □

Computing a k-K_r-Packing with $(r-2)$-Overlap

We have obtained a $2(rk - r)$ kernel for the k-K_r-Packing with $(r-2)$-Overlap problem. We can now apply a brute-force algorithm on the kernel to find a solution. First, remember that the instance was reduced by the clique-crown reduction rule (Rule 2) and the cliques in the head H becomes the set of cliques $\mathcal{F}$ in the annotated version. Therefore, we are now looking for an annotated solution in the reduced instance G' (Definition 3).

The brute-force algorithm first finds all r-cliques $\mathcal{W}$ in the reduced instance G' and after that evaluates if each possible selection of $k - |H|$ r-cliques from $\mathcal{W}$ is an annotated solution in G'.

By Claim 1, we can obtain a k-solution in the original instance combining an annotated solution in G' with the set of mapped r-cliques. The time of this brute-force algorithm is $O((2rk - 2r)^{rk})$ and the k-K_r-Packing with $(r-2)$-Overlap Problem can be decided in time $O((2rk - 2r)^{rk} + n^r)$.

We will show in Section 5 how we can apply a faster FPT-algorithm in the kernel instead of brute-force.

4 An FPT-Algorithm for the k-H-Packing with t-Overlap Problem

In this section, we introduce a fixed-parameter algorithm for the k-H-Packing with t-Overlap Problem for an arbitrary graph H and any value $0 \leq t < r$, where t is a fixed constant [18]. The formal definition of the problem studied in this section is as follows.

The k-H-Packing with t-Overlap problem
Input: A graph G and a non-negative integer k.
Parameter: k
Question: Does G contain at least k subgraphs $\mathcal{K} = \{Q_1^*, \ldots, Q_k^*\}$ where each Q_i^* is isomorphic to a graph H and $|V(Q_i^*) \cap V(Q_j^*)| \leq t$, for any pair Q_i^*, Q_j^*?

The key point of our fixed-parameter algorithm is based on the following lemma which is a generalization of Observation 2 in [6]. This lemma proves how a maximal solution intersects with a k-solution of the graph G, assuming that G has one.

Lemma 7 *Let $\mathcal{M}$ and $\mathcal{K}$ be a maximal H-Packing with t-Overlap and a k-H-Packing with t-Overlap, respectively. We claim that any $Q^* \in \mathcal{K}$ overlaps with some $Q \in \mathcal{M}$ in at least $t+1$ vertices, i.e., $|V(Q^*) \cap V(Q)| \geq t+1$. Furthermore, there is no pair $Q_i^*, Q_j^* \in \mathcal{K}$ for $i \neq j$ that overlaps in the same set of vertices with Q i.e., $V(Q_i^*) \cap V(Q) \neq V(Q_j^*) \cap V(Q)$.*

Proof: Assume by contradiction that there is an H-subgraph $Q^* \in \mathcal{K}$ such that for any H-subgraph $Q \in \mathcal{M}$, the overlap between them is at most t, i.e., $|V(Q^*) \cap V(Q)| \leq t$. However, in this case, we could add Q^* to $\mathcal{M}$, and $\mathcal{M} \cup Q^*$ is an H-Packing with t-Overlap contradicting the assumption of the maximality of $\mathcal{M}$.

To prove the second part of the lemma, assume by contradiction that there is a pair $Q_i^*, Q_j^* \in \mathcal{K}$ that overlaps in the same set of vertices for all $Q \in \mathcal{M}$. However, by the first part of the lemma, we know that there is at least one $Q \in \mathcal{M}$ such that $|V(Q_i^*) \cap V(Q)| \geq t+1$. This would imply that $|V(Q_i^*) \cap V(Q_j^*)| \geq t+1$, a contradiction since the k-H-Packing with t-Overlap problem does not allow overlap greater than t. □

Lemma 7 states that every H-subgraph of a k-solution $\mathcal{K}$ overlaps in at least $t+1$ vertices with some H-subgraph of a maximal solution $\mathcal{M}$. Let us call this intersection of $t+1$ vertices a *feasible seed*. A feasible seed is shared only by a unique pair composed of an H-subgraph of $\mathcal{M}$ and an H-subgraph of $\mathcal{K}$. Thus, a feasible seed is contained in only one H-subgraph of a k-solution $\mathcal{K}$.

Observation 2 *If the graph G has a k-H-packing with t-Overlap $\mathcal{K}$ each H-subgraph in $\mathcal{K}$ has at least one feasible seed.*

The left-side of Figure 4 shows an example of this intersection for the 4-K_4-Packing with 1-Overlap problem ($H = K_4$, $k = 4$, and $t = 1$). The two K_4's of the maximal solution are indicated by solid lines while the four K_4's of the k-solution are indicated with solid and dashed lines. Edges of the graph that do not belong to any of these solutions are indicated in light gray. The feasible seeds in the example have size $t + 1 = 2$. A collection of four feasible seeds is $\{\{1, 3\}, \{3, 4\}, \{5, 6\}, \{6, 8\}\}$; the vertices of these feasible seeds are filled in the figure.

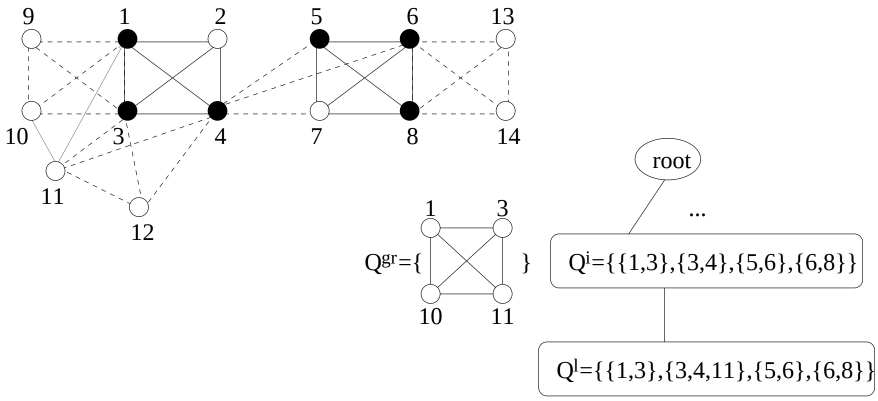

Figure 4: On the left side, an intersection of a k-solution with a maximal solution of the 4-K_4-Packing with 1-Overlap problem ($k = 4$ and $t = 1$). The seeds in this example are of size $t + 1 = 2$. To the right, part of the search tree corresponding to the instance to the left. The set Q^{gr} corresponds to a set of K_4's found by a greedy algorithm while the sets Q^i and Q^l represent k sets of vertices at different nodes of the search tree.

We now proceed to describe the algorithm for the k-H-Packing with t-Overlap problem. First, we obtain a maximal solution $\mathcal{M}$ of G. If the number of H-subgraphs in $\mathcal{M}$ is at least k then $\mathcal{M}$ is a k-H-Packing with t-Overlap and the algorithm stops. Otherwise, we want to find a k-solution using k feasible seeds (Observation 2).

Since we do not know if a set of $t + 1$ vertices of an H-subgraph $Q \in \mathcal{M}$ is a feasible seed, we would need to consider all the distinct sets of $t + 1$ vertices from $V(Q)$. To avoid confusion, we call these sets simply *seeds*. Two seeds are distinct if they differ by at least one vertex. Therefore, seeds can overlap in at most t vertices. The set of all possible seeds from all H-subgraphs of $\mathcal{M}$ is called *the universe of seeds*. Observe that there are not duplicate seeds in this universe. Otherwise, there would be at least one pair of H-subgraphs of $\mathcal{M}$ with the same seed implying that they overlap in at least $t + 1$ vertices.

Now, we create a search tree where at each node i there is a collection $\mathbf{Q^i}$ of k sets of vertices. Each set represents an H-subgraph that would be part of the k-solution. Initially, the root has a child i for each possible selection of k seeds from the universe of seeds. The collection $\mathbf{Q^i}$ is initialized with these k seeds, i.e., $\mathbf{Q^i} = \{s_1^i, \ldots, s_k^i\}$. Since we are trying all possible selections of k seeds, at least one child i should have k feasible seeds each one contained in a different H-subgraph of a k-solution, assuming G has one.

The right side of Figure 4 shows one child of the root of the search tree created with the maximal solution in the left. This child has the collection $\mathbf{Q^i} = \{\{1,3\}, \{3,4\}, \{5,6\}, \{6,8\}\}$.Observe that the seeds in $\mathbf{Q^i}$.

We say that $\mathbf{Q^i}$ *is completed into* a k-solution, if each seed in $\mathbf{Q^i}$ is completed into an H-subgraph such that any pair of H-completed subgraphs overlaps in at most t vertices. The sponsors that complete the seeds in $\mathbf{Q^i}$ are called *feasible sponsors*.

Next for each child i of the root, the goal is to try to find a feasible sponsor for each seed in $\mathbf{Q^i}$. Before explaining how we can find such sponsors, we next introduce a simple way to discard some of the sponsors that a seed $s_j^i \in \mathbf{Q^i}$ could have.

Observation 3 *Note that if there is a sponsor A of a seed s_j^i such that $s_j^i \cdot A$ overlaps in at least $t+1$ vertices with some other seed $s_l^i \in \mathbf{Q^i}$, for $s_l^i \neq s_j^i$, then the the collection $\mathbf{Q^i} = \{s_1^i, \ldots, s_j^i \cup A, \ldots, s_k^i\}$ cannot be completed into a k-solution.*

Therefore, at any step of the algorithm for a seed s_j^i we only consider a sponsor A if $s_j^i \cdot A$ overlaps in at most t vertices with every seed $s_l^i \in \mathbf{Q^i}$, for $s_l^i \neq s_j^i$.

In Figure 4, the sponsors $\{2,4\}$ and $\{5,7\}$ are not considered to complete $\{1,3\}$ and $\{6,8\}$, respectively. Discarding such sponsors ensures that the overlap between any pair of seeds is at any stage at most t.

Lemma 8 *If a seed $s_j^i \in \mathbf{Q^i}$ does not have at least one sponsor A such that $s_j^i \cdot A$ overlaps in at most t vertices with every seed $s_l^i \in \mathbf{Q^i}$, for $s_l^i \neq s_j^i$, then $\mathbf{Q^i}$ cannot be completed into a k-solution.*

Now, we explain how we attempt to complete the collection $\mathbf{Q^i}$ in a greedy fashion. Let $\mathbf{Q^{gr}}$ be the set of H-subgraphs found by a greedy algorithm at child i. Initially, $\mathbf{Q^{gr}} = \emptyset$. At iteration j, the greedy algorithm searches a sponsor A for s_j^i such that $s_j^i \cdot A$ overlaps in at most t vertices with every H-subgraph in $\mathbf{Q^{gr}}$. If such sponsor exists, greedy adds $s_j^i \cdot A$ to $\mathbf{Q^{gr}}$, i.e., $\mathbf{Q^{gr}} = \mathbf{Q^{gr}} \cup s_j^i \cdot A$; if not, greedy stops. If all the seeds of $\mathbf{Q^i}$ were completed then we have a k-H-Packing with t-Overlap.

If the greedy algorithm cannot find a k-H-Packing with t-Overlap, then the next step will be to increase the size of one of the seeds of $\mathbf{Q^i}$ by one vertex. Let s_j^i be the seed in $\mathbf{Q^i}$ that could not be completed by the greedy algorithm. Greedy could not complete s_j^i because for each sponsor A of s_j^i, the H-subgraph $s_j^i \cdot A$ overlaps in more than t vertices with at least one H-subgraph in $\mathbf{Q^{gr}}$. For example, in Figure 4, greedy completed $\{1,3\}$ with the sponsor $\{10,11\}$, and $\{1,3\} \cdot \{10,11\}$ is added to $\mathbf{Q^{gr}}$. After that greedy cannot complete $\{3,4\}$. The seed has only one sponsor $\{11,12\}$ but $\{3,4\} \cdot \{11,12\}$ overlaps in two vertices with $\{1,3\} \cdot \{10,11\}$ in $\mathbf{Q^{gr}}$. Note that the sponsor $\{1,2\}$ of the seed $\{3,4\}$ is discarded (Observation 3).

If $\mathbf{Q^i}$ can be completed into a k-solution, then at least one of the sponsors of s^i_j is feasible. We do not know which one it is, but we are certain that this feasible sponsor shares some vertices with at least one H-subgraph in $\mathbf{Q^{gr}}$. We will use this intersection of vertices to find such a feasible sponsor.

Let us denote as $I(\mathbf{Q^{gr}}, \mathsf{Sponsors}(s^i_j))$ the set of vertices that are shared between each sponsor of s^i_j and each H-subgraph in $\mathbf{Q^{gr}}$. We will increase the size of the seed s^i_j by one vertex by creating a child l of the node i for each vertex $v_l \in I(\mathbf{Q^{gr}}, \mathsf{Sponsors}(s^i_j))$.

The collection of seeds at child l, $\mathbf{Q^l}$ is the same as the collection of its parent i with the update of the seed s^i_j as $s^i_j \cup v_l$, i.e., $\mathbf{Q^l} = \{s^i_1, \ldots, s^i_{j-1}, s^i_j \cup v_l, s^i_{j+1}, \ldots, s^i_k\}$. After that, the greedy algorithm is repeated at the collection $\mathbf{Q^l}$ of child l ($\mathbf{Q^{gr}}$ starts empty again). In the example of Figure 4, there is one child of the node i where the seed $\{3,4\}$ is updated with the vertex 11. Observe that after this update, the sponsor $\{10,11\}$ is discarded to complete $\{1,3\}$ (Observation 3).

The algorithm stops attempting to complete $\mathbf{Q^i}$ or some collection of a descendant of the node i when there are no sponsors for one of the seeds (Lemma 8). Otherwise, one of the leaves of the tree would have a k-H-Packing with t-Overlap. In the example of Figure 4, one leaf of the search tree would complete the collection into the solution $\mathcal{K} = \{\{1,3\} \cdot \{9,10\}, \{3,4\} \cdot \{11,12\}, \{5,6\} \cdot \{4,7\}, \{6,8\} \cdot \{13,14\}\}$.

The pseudocode of the algorithm is shown in Algorithms 2 and 3.

Algorithm 2 BST-k-H-Packing with t-Overlap Algorithm

Input: A graph $G = (V, E)$ and a non-negative integer k.
$solution = \emptyset$, $i = 1$
Compute a maximal H-packing with t-Overlap $\mathcal{M}$ in G
if $|\mathcal{M}| \geq k$ **then**
  Accept
else
  while $i \leq |\binom{\mathsf{UniverseOfSeeds}}{k}|$ and $solution = \emptyset$ **do**
    Let $\mathbf{Q^i} = \{s^i_1, s^i_2, \ldots, s^i_k\}$ be the i-th combination of $\binom{\mathsf{UniverseOfSeeds}}{k}$
    CreateNode(root,$\mathbf{Q^i}$) {*Create node with parent root and collection* $\mathbf{Q^i}$ }
    $solution$ = Completion(node i,$\mathbf{Q^i}$)
    $i = i + 1$
  end while
  return $solution$
end if

Algorithm 3 COMPLETION(NODE i,$\mathbf{Q^i}$)

$\mathbf{Q^{gr}}$=GREEDY($\mathbf{Q^i}$) {*Greedily complete the collection* $\mathbf{Q^i}$ }
{*The greedy algorithm returns* $\mathbf{Q^{gr}} = \emptyset$ *when* $\mathbf{Q^i}$ *does not have a solution (Lemma 8)* }
if $\mathbf{Q^{gr}}$ contains k H-subgraphs OR $\mathbf{Q^{gr}} = \emptyset$ **then**
  return $\mathbf{Q^{gr}}$
else
  Let s^i_j be the first seed of $\mathbf{Q^i}$ not completed by GREEDY
  $l = 1$, $solution = \emptyset$
  while $l \leq |I(\mathbf{Q^{gr}}, \mathsf{Sponsors}(s^i_j))|$ and $solution = \emptyset$ **do**
    $\mathbf{Q^l} = \{s^i_1, \ldots, s^i_j \cup v_l, \ldots, s^i_k\}$
    CREATENODE(NODE l,$\mathbf{Q^l}$)
    $solution$ = COMPLETION(NODE l,$\mathbf{Q^l}$)
    $l = l + 1$
  end while
  return $solution$
end if

4.1 Correctness

The next basic lemma will help us to prove that the algorithm is correct.

Lemma 9 *If A is a sponsor of the seed s^i_j then $A \backslash X$ is a sponsor of $s^i_j \cup X$, for any $X \subset A$.*

We next prove the correctness of the algorithm.

Theorem 1 *If the graph G has a k-solution $\mathcal{K} = \{Q^*_1, \ldots, Q^*_k\}$ then this solution will be propagated in at least one path from the root to a leaf of the tree.*

Proving the theorem is equivalent to proving the following claim.

Claim 8 *There is a path $P =< i_1, i_2, \ldots, i_m >$ such that each node i_l in P has the collection $\mathbf{Q^{i_l}} = \{s^{i_l}_1, \ldots, s^{i_l}_k\}$ where $s^{i_l}_j \subseteq V(Q^*_j)$ for $1 \leq j \leq k$.*

Proof:

We prove this claim by induction on the number of levels.

Since we are creating a child of the root for each possible selection of k seeds from the universe of seeds, there would be at least one child of the root with collection $\mathbf{Q^{i_1}} = \{s^{i_l}_1, \ldots, s^{i_l}_k\}$ where the claim will follow.

By Observation 2, each feasible seed in $\mathbf{Q^i}$ has at least one feasible sponsor. Let $\{A^*_1, \ldots, A^*_k\}$ be the set of feasible sponsors where A^*_j is the feasible sponsor of s^i_j. That is, the k-solution can be seen as $\mathcal{K} = \{s^i_1 \cdot A^*_1, \ldots, s^i_k \cdot A^*_k\}$.

Next we show that for the remaining nodes of P, the seeds are updated only with vertices from the feasible sponsors.

Let us suppose that the greedy algorithm failed to complete a seed $s_j^{i_1}$ at level 1. The seed $s_j^{i_1}$ has at least one feasible sponsor A_j^*. Greedy failed to complete $s_j^{i_1}$, if the H-subgraph formed with $s_j^{i_1}$ and each sponsor of $s_j^{i_1}$ (including the feasible one A_j^*) overlaps in more than t vertices with an H-subgraph completed by greedy (i.e., an H-subgraph in the set $\mathbf{Q^{gr}}$). Therefore at level 2, there is at least one child of the node i_1 where the seed $s_j^{i_1}$ is updated with one vertex from the feasible sponsor A_j^*. That is, $\mathbf{Q^{i_2}} = \{s_1^{i_1}, \ldots, s_j^{i_1} \cup v^*, \ldots, s_k^{i_1}\}$ where $v^* \in A_j^*$, and the claim follows.

Now, let us assume that the claim is true up to the level $h-1$. We next show that claim holds for level h. Let $< i_2, \ldots, i_{h-1} >$ be the subpath of P where one seed in each node is updated with one vertex from its feasible sponsor.

Suppose by contradiction that at level $h-1$, the greedy algorithm could not complete the seed $s_j^{i_{h-1}}$ but there is no child of the node i_{h-1} such that $s_j^{i_{h-1}}$ is updated with a vertex from its feasible sponsor.

Let us suppose that U^* is the set of vertices that has been added to $s_j^{i_1}$ during the levels $1, \ldots, h-1$. By our assumption, the seed $s_j^{i_1}$ is feasible, and it has been updated only with vertices from its feasible sponsor. Therefore, $U^* \subset A_j^*$.

By Lemma 9, $A_j^* \backslash U^*$ is a sponsor of $s_j^{i_{h-1}}$. Thus, the only way that none of the children of i_{h-1} would update $s_j^{i_{h-1}}$ with a vertex from $A_j^* \backslash U^*$ is if $A_j^* \backslash U^*$ is not a sponsor of $s_j^{i_{h-1}}$.

However, this would imply that the H-subgraph $s_j^{i_{h-1}} \cdot (A_j^* \backslash U^*) = s_j^{i_1} \cdot A_j^*$ overlaps in more than t vertices with some feasible seed $s_l^{i_1} \in \mathbf{Q^{i_1}}$. This contradicts our assumption that the collection $\mathbf{Q^{i_1}}$ can be completed into a k-solution.

□

4.2 Analysis

Lemma 10 *The root has at most* $\left(\frac{e^2 r}{t+1}\right)^{k(t+1)}$ *children.*

Proof: There are $\binom{|V(H)|}{t+1}$ distinct sets of $t+1$ vertices (seeds) in the set of vertices of an H-subgraph H. The universe of seeds was defined as the set of all possible sets of $t+1$ vertices in $V(H)$ for each $H \in \mathcal{M}$. Since $|\mathcal{M}| \leq k-1$, and $|V(H)| = r$ then there are at most $(k-1)\left(\frac{er}{t+1}\right)^{t+1}$ seeds in the universe of seeds.

From the root of the search tree, we create a node i for each possible selection of k seeds of the universe of seeds, i.e., $\binom{|universe of seeds|}{k}$. Hence, $\binom{(k-1)\left(\frac{er}{t+1}\right)^{t+1}}{k}$ $\leq \left(\frac{e(k-1)\left(\frac{er}{t+1}\right)^{t+1}}{k}\right)^k \leq \left(\frac{e^2 r}{t+1}\right)^{k(t+1)}$. □

Lemma 11 *The height of the search tree is at most* $(r-t-1)k-1$.

Proof: If the greedy algorithm cannot complete the collection $\mathbf{Q^i}$ of a node i, then we create at least one child of i. In this new child, the first seed not completed by greedy, let's say s^i_j, is updated with one vertex. Since at the first level $|s^i_j| = t+1$, then s^i_j could be completed into an H-subgraph in at most $r-(t+1)$ levels (which are not necessarily consecutive).

At most $k-1$ H-subgraphs are completed in this way since the last k H-subgraph is completed in greedy fashion. Therefore, we need at most $(r-t-1)k-1$ levels to complete the k seeds in $\mathbf{Q^i}$. □

Lemma 12 *A node i at level h can have at most $r(k-1)$ children if $h \leq (r-t-2)k$ and at most $r(k-m-1)$ where $m = h-(r-t-2)k$ if $h > (r-t-2)k$.*

Proof: There is a child of i for each vertex in $I(\mathbf{Q^{gr}}, \mathsf{Sponsors}(s^i_j))$, where s^i_j is the first seed not completed by greedy. This is the set of vertices shared by each sponsor of s^i_j with the H-subgraphs completed by greedy, i.e., the set $\mathbf{Q^{gr}}$. Therefore, $I(\mathbf{Q^{gr}}, \mathsf{Sponsors}(s^i_j)) \leq r|\mathbf{Q^{gr}}|$.

The greedy algorithm needs to complete only the seeds in $\mathbf{Q^i}$ that are not already completed H-subgraphs. Therefore, $|\mathbf{Q^{gr}}| \leq |\mathbf{Q^i}| - m$ where m is the number of seeds of $\mathbf{Q^i}$ that are completed H-subgraphs.

Assuming all the seeds of $\mathbf{Q^i} - m$ were completed by greedy but the last one, i.e., $|\mathbf{Q^{gr}}| \leq |\mathbf{Q^i}| - m - 1$, then $I(\mathbf{Q^{gr}}, \mathsf{Sponsors}(s^i_j)) \leq r(|\mathbf{Q^i}| - m - 1) \leq r(k-m-1)$.

Now, we need to determine how many seeds of $\mathbf{Q^i}$ are already H-subgraphs at level h, i.e., the value of m. Since a seed s^i_j can be completed into an H-subgraph in at most $r-(t+1)$ levels but these are not necessarily consecutive, then we cannot guarantee that $\frac{h}{r-(t+1)}$ is the number of H-subgraphs at level h. Therefore, in the worst-case one vertex is added to each seed of $\mathbf{Q^i}$ level by level. In this way, in at most $(r-t-2)k$ levels every seed of $\mathbf{Q^i}$ could have $r-1$ vertices, and at level $(r-t-2)k+1$ we could obtain the first seed completed into an H-subgraph. After that level, the remaining seeds of $\mathbf{Q^i}$ are completed into H-subgraphs by adding one vertex.

□

Theorem 2 *The k-H-Packing with t-Overlap can be solved in $O(r^{rk}k^{(r-t-1)k+2}n^r)$ time.*

Proof: By Lemmas 11 and 12, the size of the tree is

$$\left(\frac{e^2 r}{t+1}\right)^{k(t+1)} \prod_{i=1}^{(r-t-2)k} r(k-1) + \prod_{i=1}^{k-r+t} r(k-i)$$
$$< \left(\frac{e^2 r}{t+1}\right)^{k(t+1)} (r(k-1))^{(r-t-1)k-1}.$$

A maximal solution $\mathcal{M}$ can be computed in time $O(krn^r)$, which is also the required time to compute the list of sponsors of the seeds. The greedy algorithm runs in $O(k^2 rn^r)$.

□

5 An FPT-algorithm for the k-K_r-Packing with $(r-2)$-Overlap Problem

In this section, we will show how the analysis of the FPT-algorithm of Section 4 is substantially simplified when $H = K_r$ and $t = r-2$, i.e., the k-K_r-Packing with $(r-2)$-Overlap problem. After that we obtain the running time of this algorithm when is applied to the kernel of Subsection 3.3. We show that in this way the k-K_r-Packing with $(r-2)$-Overlap problem can be solved faster than with a brute-force search on the kernel.

Let us begin with a finer analysis of the search tree for the particular case when $H = K_r$ and $t = r-2$. For that, we need to determine: the number of children of the root, the height of the tree, and the number of children that any node can have at level h of the tree. The number of children of the root is bounded by $\binom{r^2(k-1)}{k} = O(r^{2k}k^k)$. Since the size of each seed at level 0 is $t+1 = (r-2)+1 = r-1$, then each seed is completed into an r-clique in one level. Therefore, the height of the tree is at most $k-1$ (the last seed is completed in greedy fashion). At level h, there at least h seeds that have been completed into an r-clique. This follows because each seed only needs one vertex to be completed into an r-clique, and in each level one seed is updated with one vertex. For this reason, each node can have at most $r(k-h)$ children at level h.

Combining these values, the size of the search tree is given by:

$$r^{2k}k^k \prod_{h=1}^{k-1} r(k-h) < r^{2k}k^k r^{k-1}k^k = r^{3k-1}k^{2k}$$

The time spent in each node is $O(k^2rn^r)$. Therefore, the running time of the FPT-algorithm for the k-K_r-Packing with $(r-2)$-Overlap problem applied to the original instance is $O(r^{3k-1}k^{2(k+r)}n^r))$.

Applying this algorithm to the $O(2(rk-r))$ kernel for the k-K_r-Packing with $(r-2)$-Overlap Problem obtained in Subsection 3.3, we have the following running time.

Theorem 3 *The k-K_r-Packing with $(r-2)$-Overlap Problem can be solved in $O(r^{3k+r-1}k^{2k+3r} + n^r)$ time.*

6 Conclusions

We have introduced the k-$\mathcal{H}$-Packing with t-Overlap problem to overcome the deficiencies of previous work on the community discovering problem. We designed a global reduction rule, the clique-crown decomposition, for the k-H-Packing with t-Overlap problem, when $H = K_r$. Using our reduction rule, we achieved reductions to a kernel for this problem when $t = r-2$. We emphasize that the clique-crown reduction rule can be extended to consider other families of graphs as well. For example, it would be interesting to consider community models less restrictive than cliques such as s-cliques, s-clubs, and s-plexes.

When computing the clique-crown decomposition in Section 5, it is assumed that cliques in O are vertex-disjoint and cliques in $\mathsf{Cliques}(O)$ overlap in at most $t+1$. If these two conditions do not follow then a perfect matching would not guarantee that a pair of mapped r-cliques overlap in at most t vertices. Therefore, it remains how to design a different injective function that satisfies this overlap condition.

In addition, we developed the first fixed-parameter algorithm for the the k-H-Packing with t-Overlap problem for any graph H and any value $t < r$. We also provided a finer analysis of this algorithm for the case when $H = K_r$ and $t = r - 2$, and we applied this algorithm to the kernel obtained for the k-K_r-Packing with $(r-2)$-Overlap problem. This approach is faster than applying a brute-force algorithm on the kernel.

References

[1] H. Balakrishman and N. Deo. Detecting communities using bibliographic metrics. In *Proceedings of the IEEE International Conference on Granular Computing*, pages 293–298. IEEE Computer Society, 2006. doi:10.1109/GRC.2006.1635799.

[2] B. Chor, M. Fellows, and D. Juedes. Linear kernels in linear time, or how to save k colors in $\mathcal{O}(n^2)$ steps. In J. Hromkovič, M. Nagl, and B. Westfechtel, editors, *Proceedings of the 30th Workshop on Graph Theoretic Concepts in Computer Science (WG 2004)*, volume 3353, pages 257–269. Springer, Heidelberg, 2004. doi:10.1007/978-3-540-30559-0_22.

[3] P. Damaschke. Fixed-parameter tractable generalizations of cluster editing. In T. Calamoneri, I. Finocchi, and G. F. Italiano, editors, *The 6th International Conference on Algorithms and Complexity (CIAC)*, volume 3998, pages 344–355. Springer Berlin Heidelberg, 2006. doi:10.1007/11758471_33.

[4] M. G. Everett and S. P. Borgatti. Analyzing clique overlap. *Connections*, 21(1):49–61, 1998.

[5] M. Fellows, J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann. Graph-based data clustering with overlaps. In H. Q. Ngo, editor, *Computing and Combinatorics, 15th Annual International Conference, COCOON 2009*, volume 5609, pages 516–526. Springer Berlin Heidelberg, 2009. doi:10.1007/978-3-642-02882-3_51.

[6] M. Fellows, P. Heggernes, F. Rosamond, C. Sloper, and J. Telle. Finding k disjoint triangles in an arbitrary graph. In J. Hromkovič, M. Nagl, and B. Westfechtel, editors, *The 30th Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 3353, pages 235–244. Springer Berlin Heidelberg, 2004. doi:10.1007/978-3-540-30559-0_20.

[7] M. Fellows, C. Knauer, N. Nishimura, P. Ragde, F. Rosamond, U. Stege, D. Thilikos, and S. Whitesides. Faster fixed-parameter tractable algorithms for matching and packing problems. *Algorithmica*, 52(2):167–176, 2008. doi:10.1007/s00453-007-9146-y.

[8] M. Girvan and M. Newman. Community structure in social and biological networks. In *Proceedings of the National Academy of Science of the United States of America*, volume 99, pages 7821–7826, 2002. doi:10.1073/pnas.122653799.

[9] S. Hartung, C. Komusiewicz, and A. Nichterlein. On structural parameterizations for the 2-club problem. In P. van Emde Boas, F. C. A. Groen, G. F. Italiano, J. Nawrocki, and H. Sack, editors, *Proceedings of the 39th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 7741 of *LNCS*, pages 233–243. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-642-35843-2_21.

[10] D. Kirkpatrick and P. Hell. On the completeness of a generalized matching problem. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 240–245. ACM, 1978. doi:10.1145/800133.804353.

[11] C. Komusiewicz and M. Sorge. Finding dense subgraphs of sparse graphs. In D. M. Thilikos and G. J. Woeginger, editors, *Parameterized and Exact Computation - 7th International Symposium (IPEC)*, volume 7535, pages 242–251. Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-33293-7_23.

[12] F. Luo, J. Z. Wang, and E. Promislow. Exploring local community structures in large networks. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 233–239. IEEE Computer Society, 2006. doi:10.1109/WI.2006.72.

[13] L. Mathieson, E. Prieto, and P. Shaw. Packing edge disjoint triangles: A parameterized view. In R. Downey, M. Fellows, and F. Dehne, editors, *Proceedings of the First International Workshop on Parameterized and Exact Computation (IWPEC 2004)*, volume 3162, pages 127–137. Springer Berlin Heidelberg, 2004. doi:10.1007/978-3-540-28639-4_12.

[14] H. Moser. A problem kernelization for graph packing. In M. Nielsen, A. Kučera, P. B. Miltersen, C. Palamidessi, P. Tůma, and F. Valencia, editors, *Proceedings of the 35th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 5404, pages 401–412. Springer Berlin Heidelberg, 2009. doi:10.1007/978-3-540-95891-8_37.

[15] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005. doi:10.1038/nature03607.

[16] E. Prieto and C. Sloper. Looking at the Stars. *Theoretical Computer Science*, 351(3):437–445, 2006. doi:10.1016/j.tcs.2005.10.009.

[17] J. Romero and A. López-Ortiz. The $\mathcal{G}$-packing with t-overlap problem. In S. P. Pal and K. Sadakane, editors, *Proceedings of the 8th International Workshop on Algorithms and Computation (WALCOM)*, volume 8344, pages 114–124. Springer Berlin Heidelberg, 2014. doi:10.1007/978-3-319-04657-0_13.

[18] J. Romero and A. López-Ortiz. A parameterized algorithm for packing overlapping subgraphs. In E. A. Hirsch, S. O. Kuznetsov, J. Éric Pin, and N. K. Vereshchagin, editors, *Proceedings of the 9th International Computer Science Symposium in Russia (CSR)*, volume 8476, pages 325–336. Springer Berlin Heidelberg, 2014. doi:10.1007/978-3-319-06686-8_25.

[19] A. Schäfer, C. Komusiewicz, H. Moser, and R. Niedermeier. Parameterized computational complexity of finding small-diameter subgraphs. *Optimization Letters*, 6(5):883–891, 2012. doi:10.1007/s11590-011-0311-5.