

Hardness and Fixed Parameter Tractability for Pinwheel Scheduling Problems

Yusuke Kobayashi 

Research Institute for Mathematical Sciences, Kyoto University, Japan

Bingkai Lin 

School of Computer Science, Nanjing University, China

Abstract

In the PINWHEEL PACKING problem, we are given a set of recurring tasks, each associated with a positive integer a_i for task i . The objective is to select one task to perform each day such that every task i is performed at least once within every a_i consecutive days. The exact computational complexity of this problem, where $\sum 1/a_i = 1$, has remained an open question for more than 30 years; in particular, it is still unknown whether the problem is NP-hard. The first contribution of this paper is to show that PINWHEEL PACKING cannot be solved in polynomial time under a standard complexity assumption, improving upon the hardness result shown by Jacobs and Longo. Additionally, we present fixed-parameter algorithms for variants of PINWHEEL PACKING, parameterized by the number of tasks.

2012 ACM Subject Classification Theory of computation $\rightarrow$ Design and analysis of algorithms; Theory of computation $\rightarrow$ Computational complexity and cryptography

Keywords and phrases Pinwheel Scheduling, Polynomial-time Solvability, Packing and Covering, Fixed Parameter Algorithms

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2025.47

Funding *Yusuke Kobayashi*: JSPS KAKENHI Grant Numbers JP22H05001 and JP24K02901.

1 Introduction

1.1 Pinwheel Scheduling

The (packing version of) *Pinwheel Scheduling* problem was introduced by Holte et al. [12] in 1989 as a formalization of a real-time scheduling problem. In the problem, we are given k recurring tasks and a positive integer a_i for each task $i \in [k] = \{1, \dots, k\}$. The objective is to select one task to perform each day so that each task i is performed at least once every a_i days. In other words, we want to find pairwise disjoint sets $S_1, \dots, S_k \subseteq \mathbb{Z}$ such that

$$|[m, m + a_i) \cap S_i| \geq 1 \quad (1)$$

for all $i \in [k]$ and $m \in \mathbb{Z}$, where S_i represents the set of days on which task i is performed. Note that each S_i can be an infinite set. A natural computational problem is to determine the existence of such sets $S_1, \dots, S_k \subseteq \mathbb{Z}$, which is formally described as follows. Note that we call this problem PINWHEEL PACKING in this paper, because we will discuss the corresponding *covering* variant later.

PINWHEEL PACKING

Input: A sequence of positive integers $A = (a_i)_{i \in [k]}$.

Task: Determine whether there exist pairwise disjoint sets $S_1, \dots, S_k \subseteq \mathbb{Z}$ such that $|[m, m + a_i) \cap S_i| \geq 1$ for all $i \in [k]$ and $m \in \mathbb{Z}$.



© Yusuke Kobayashi and Bingkai Lin;

licensed under Creative Commons License CC-BY 4.0

36th International Symposium on Algorithms and Computation (ISAAC 2025).

Editors: Ho-Lin Chen, Wing-Kai Hon, and Meng-Tsung Tsai; Article No. 47; pp. 47:1–47:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Despite the simplicity and naturalness of the problem setting, little is known about the computational complexity of this problem. It has been shown in [12, Corollary 2.2] that PINWHEEL PACKING is in PSPACE; however, it is unknown whether this problem is in NP. Furthermore, it is unknown whether this problem is NP-hard. See [19] for details.

Since each task i has to be performed on at least a $\frac{1}{a_i}$ fraction of a sufficiently long consecutive sequence of days, if $A = (a_i)_{i \in [k]}$ is a yes-instance of PINWHEEL PACKING, then $\sum_{i=1}^k \frac{1}{a_i}$ (called the *density* of A) is at most 1. An instance $A = (a_i)_{i \in [k]}$ is called *dense* if its density is exactly 1. As we see in Section 1.2, for dense instances, PINWHEEL PACKING is equivalent to several other problems, highlighting that dense instances form an interesting special case. However, the exact complexity of dense instances of PINWHEEL PACKING has been open for more than 30 years. In fact, it is noted in [12, Section 4] that the NP-hardness of this problem remains an open question, and it is explicitly conjectured by Kawamura and Soejima [21, Conjecture 19] that PINWHEEL PACKING for dense instances is NP-complete. Note that the NP-hardness has been shown when the input sequence is represented in a concise notation; see [12, Theorem 4.12] and [19, Theorem 5]. Jacobs and Longo showed in their preprint [16] that (dense instances of) PINWHEEL PACKING cannot be solved in polynomial time unless SAT with n variables has a randomized algorithm running in expected time $n^{O(\log n \log \log n)}$.

1.2 Exact Packing and Covering

As mentioned in the previous subsection, for dense instances, PINWHEEL PACKING is equivalent to several other problems.

Observe that, if $S_1, \dots, S_k \subseteq \mathbb{Z}$ form a solution for a dense instance $A = (a_i)_{i \in [k]}$ of PINWHEEL PACKING, then they satisfy (1) with equality. Using this observation, for dense instances, PINWHEEL PACKING is equivalent to the following *exact* variant.

EXACT PINWHEEL PACKING

Input: A sequence of positive integers $A = (a_i)_{i \in [k]}$.

Task: Determine whether there exist pairwise disjoint sets $S_1, \dots, S_k \subseteq \mathbb{Z}$ such that $|[m, m + a_i) \cap S_i| = 1$ for all $i \in [k]$ and $m \in \mathbb{Z}$.

Note that the desired condition of this problem means that each task i is performed *exactly* once every a_i days. Therefore, S_i satisfies $|[m, m + a_i) \cap S_i| = 1$ for all $m \in \mathbb{Z}$ if and only if it is a *residue class* modulo a_i , i.e., $S_i = \{r_i + a_i \cdot q : q \in \mathbb{Z}\}$ for some integer r_i . Independently of its connection to PINWHEEL PACKING, EXACT PINWHEEL PACKING is an interesting problem in its own right. Indeed, EXACT PINWHEEL PACKING has been studied under the name *(1-server) periodic maintenance problem* [3, 25, 30] in the context of scheduling problems, and its solution has also garnered attention in discrete mathematics under the name *disjoint residue classes* [15, 21, 28]. It is shown in [3, 16, 21, 25] that EXACT PINWHEEL PACKING is NP-complete, where we note that the input is not necessarily dense.

We can also consider a problem that is, in a sense, dual to (EXACT) PINWHEEL PACKING, which is called (EXACT) PINWHEEL COVERING [19]. Consider a scenario where a single task must be performed every day, and it is to be handled by k agents. Each agent $i \in [k]$ is assigned a positive integer a_i , which represents the condition that the agent can perform the task at most once (or exactly once) every a_i days. In this setting, the problem of determining the assignment of agents corresponds to finding a collection of sets $S_1, \dots, S_k \subseteq \mathbb{Z}$ such that $\bigcup_{i \in [k]} S_i = \mathbb{Z}$ and

$$|[m, m + a_i) \cap S_i| \leq 1 \quad (\text{or } |[m, m + a_i) \cap S_i| = 1)$$

for all $i \in [k]$ and $m \in \mathbb{Z}$, where S_i represents the set of days on which agent i performs the task. That is, the problem is formally described as follows.

PINWHEEL COVERING (resp. EXACT PINWHEEL COVERING)

Input: A sequence of positive integers $A = (a_i)_{i \in [k]}$.

Task: Determine whether there exist sets $S_1, \dots, S_k \subseteq \mathbb{Z}$ such that $\bigcup_{i \in [k]} S_i = \mathbb{Z}$ and $|[m, m + a_i) \cap S_i| \leq 1$ (resp. $|[m, m + a_i) \cap S_i| = 1$) for all $i \in [k]$ and $m \in \mathbb{Z}$.

PINWHEEL COVERING was introduced and studied under the name *point patrolling* in [21]. Similarly to the packing problem, this problem is in PSPACE, and it is unknown whether this problem is NP-hard. In contrast, very recently, it was shown that EXACT PINWHEEL COVERING is NP-hard [20]. Note that feasible solutions for EXACT PINWHEEL COVERING are known as *Erdős covering systems* and have been actively studied by number theorists; see [2, 14].

Since each agent i can perform a task on at most a $\frac{1}{a_i}$ fraction of a sufficiently long consecutive sequence of days, if $A = (a_i)_{i \in [k]}$ is a yes-instance of PINWHEEL COVERING, then its density $\sum_{i=1}^k \frac{1}{a_i}$ is at least 1. Similarly to the packing problem, if the density is exactly 1, then PINWHEEL COVERING and EXACT PINWHEEL COVERING are equivalent. Furthermore, if the density of $A = (a_i)_{i \in [k]}$ is exactly 1 and $S_1, \dots, S_k \subseteq \mathbb{Z}$ satisfy $|[m, m + a_i) \cap S_i| = 1$ for all $i \in [k]$ and $m \in \mathbb{Z}$, then we see that $\bigcup_{i \in [k]} S_i = \mathbb{Z}$ if and only if $S_1, \dots, S_k$ are pairwise disjoint. By combining these facts, we arrive at the following observation.

► **Observation 1** (see [19, Section 5]). *For instances with density exactly 1, the four above problems, i.e., (EXACT) PINWHEEL PACKING and (EXACT) PINWHEEL COVERING, coincide.*

For brevity, PINWHEEL PACKING, when restricted to dense instances, is referred to as DENSE PINWHEEL PACKING. Then, Observation 1 implies that DENSE PINWHEEL PACKING is a special case of all the four above problems.

1.3 Our Contributions

As described in Section 1.1, it was shown by Jacobs and Longo [16] that DENSE PINWHEEL PACKING cannot be solved in polynomial time unless SAT with n variables has a randomized algorithm running in expected time $n^{O(\log n \log \log n)}$. The first contribution of this paper is to strengthen this result by showing that DENSE PINWHEEL PACKING cannot be solved in polynomial time under a weaker complexity assumption, partially addressing the open question in [12, Section 4] and [21, Conjecture 19].

► **Theorem 2.** DENSE PINWHEEL PACKING *cannot be solved in polynomial time unless* $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$.

Here, $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$ means that every problem in NP can be solved in $n^{O(\log n)}$ time, where n is the input size of the problem. The assumption $\text{NP} \not\subseteq \text{DTIME}(n^{O(\log n)})$ is weaker than the assumption in [16] in two key aspects: the running time is improved from $n^{O(\log n \log \log n)}$ to $n^{O(\log n)}$, and our assumption does not rely on randomization. Note that the assumption $\text{NP} \not\subseteq \text{DTIME}(n^{O(\log n)})$ is weaker than the Exponential Time Hypothesis (ETH) and $\text{NP} \not\subseteq \text{QP}$ (see Corollary 10), while it is stronger than $\text{P} \neq \text{NP}$. Theorem 2 implies that PINWHEEL PACKING and PINWHEEL COVERING cannot be solved in polynomial time under the same assumption.

► **Corollary 3.** *Neither PINWHEEL PACKING nor PINWHEEL COVERING can be solved in polynomial time unless* $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$.

In our proof for Theorem 2, we refine the argument in [16]. We first prove that EXACT PINWHEEL PACKING is hard even when $\text{lcm}(A)$ is not particularly large (see Theorem 6), where $\text{lcm}(A)$ denotes the least common multiple of $a_1, \dots, a_k$ for $A = (a_i)_{i \in [k]}$. We then reduce EXACT PINWHEEL PACKING to DENSE PINWHEEL PACKING by appending multiple copies of integer $\text{lcm}(A)$, thereby proving the hardness of DENSE PINWHEEL PACKING. The primary technical challenge lies in ensuring that $\text{lcm}(A)$ does not become excessively large.

To achieve this, Jacobs and Longo [16] reduce an NP-hard problem of size n to a new problem called PARTIAL CODING, and then further reduce it to EXACT PINWHEEL PACKING. These reductions show that EXACT PINWHEEL PACKING is hard even when $\text{lcm}(A) = n^{O(\log n \log \log n)}$. In contrast, we directly and carefully reduce an NP-hard problem to EXACT PINWHEEL PACKING, which enables us to achieve a tighter bound of $\text{lcm}(A) = n^{O(\log n)}$. To derandomize our reduction step, we utilize a sophisticated tool known as (n, k) -universal sets [26] in a non-trivial manner, which constitutes the most technical aspect of our proof. For details, see Section 2.

Theorem 2 suggests that (EXACT) PINWHEEL PACKING and (EXACT) PINWHEEL COVERING are unlikely to be solvable in polynomial time. It is then natural to consider the fixed-parameter tractability of these problems. Indeed, Gąsieniec et al. [10] showed that PINWHEEL PACKING can be solved in FPT time when parameterized by k . Our second contribution is to show that the remaining three variants can also be solved in FPT time with respect to the parameter k .

► **Theorem 4.** PINWHEEL COVERING and EXACT PINWHEEL COVERING can be solved in $O(n) + k^{O(2^k)}$ time, where $n := \sum_{i=1}^k \log a_i$ is the input size of the problem.

► **Theorem 5.** EXACT PINWHEEL PACKING can be solved in $(\log k)^{O(k^2)} \cdot n^{O(1)}$ time, where $n := \sum_{i=1}^k \log a_i$ is the input size of the problem.

We here describe proof ideas for these theorems, while formal proofs are given in Section 3. Our proof for Theorem 4 is based on the following intuition: if some a_i is extremely large, then removing a_i does not affect the feasibility of the problems. Our technical contribution lies in formalizing this intuition. We then show that it is sufficient to consider the case where each a_i is bounded by a function of k , leading to FPT algorithms.

It should be noted that this argument does not apply to EXACT PINWHEEL PACKING, as removing a large a_i could affect the feasibility of the problem. For instance, if A consists of two large integers a_1 and a_2 that are coprime, then A is a no-instance. However, removing either a_1 or a_2 would result in a yes-instance. Therefore, we adopt a completely different approach to obtain Theorem 5. We observe that a solution to EXACT PINWHEEL PACKING consists of sets $S_i = \{r_i + a_i \cdot q : q \in \mathbb{Z}\}$, where each set is determined by an integer r_i . Treating $r_1, \dots, r_k$ as integer variables, we show that the constraints for r_i can be formulated as an integer linear program by introducing several auxiliary variables. Since the integer linear programming problem can be solved in FPT time, parameterized by the number of variables, this yields an FPT algorithm for EXACT PINWHEEL PACKING.

Moreover, to highlight the similarities and differences between PINWHEEL PACKING and (EXACT) PINWHEEL COVERING, an FPT algorithm for PINWHEEL PACKING with an explicit running time bound is provided in Appendix A.

1.4 Related Work

A generalization of PINWHEEL PACKING has been studied under the name of WINDOWS SCHEDULING PROBLEM. While PINWHEEL PACKING can be viewed as a scheduling problem involving a single machine where all tasks have the same size, WINDOWS SCHEDULING PROBLEM extends this to a setting with multiple machines and tasks of varying sizes [4, 5, 16, 22].

BAMBOO GARDEN TRIMMING problem is an optimization variant of PINWHEEL PACKING. Here, the objective is to find the smallest positive number h such that $(h \cdot a_i)_{i \in [k]}$ is a yes-instance of PINWHEEL PACKING. Hardness results and approximation algorithms have been investigated for this problem [9, 11, 29].

For PINWHEEL PACKING, there has been extensive research on sufficient conditions for the density of the input A to be a yes-instance. It is not so difficult to show that A is a yes-instance if its density is at most $\frac{1}{2}$ [12]. A central question in this area is whether the same statement holds when $\frac{1}{2}$ is replaced with a larger number α . The maximum such number α is called the *density threshold*. Since $A = (2, 3, m)$ is a no-instance with a density of $\frac{5}{6} + \frac{1}{m}$ for any integer m , the density threshold is at most $\frac{5}{6}$. Building on extensive research on the density threshold [6, 7, 10, 24], Kawamura [18] recently showed that the density threshold is exactly $\frac{5}{6}$, resolving the conjecture proposed by Chan and Chin [6]. A threshold depending on the smallest element in A is investigated in [11]. It is also shown that if A consists of at most two distinct values and its density is at most 1, then it is a yes-instance [13].

2 Hardness Results

The objective of this section is to show the hardness of DENSE PINWHEEL PACKING (Theorem 2). To achieve this, we reduce a well-known NP-hard problem called 3-COLORING IN GRAPHS WITH MAXIMUM DEGREE FOUR; see [8]. In 3-COLORING IN GRAPHS WITH MAXIMUM DEGREE FOUR, the input is a graph $G = (V, E)$ with maximum degree four and the objective is to decide whether G is 3-colorable, that is, to decide whether we can color V with three colors so that adjacent vertices have different colors.

We first show in Section 2.1 that 3-COLORING IN GRAPHS WITH MAXIMUM DEGREE FOUR with n vertices can be reduced to EXACT PINWHEEL PACKING with $\text{lcm}(A) = n^{O(\log n)}$. Recall that $\text{lcm}(A)$ denotes the least common multiple of $a_1, \dots, a_k$ for $A = (a_i)_{i \in [k]}$. Then, in Section 2.2, we reduce EXACT PINWHEEL PACKING with $\text{lcm}(A) = n^{O(\log n)}$ to DENSE PINWHEEL PACKING of size $n^{O(\log n)}$.

2.1 Reduction to Exact Pinwheel Packing

► **Theorem 6.** *Let $G = (V, E)$ be a graph with $|V| = n$ whose maximum degree is four. In $n^{O(1)}$ time, we can construct a sequence of positive integers A with $\text{lcm}(A) = n^{O(\log n)}$ such that $G = (V, E)$ is 3-colorable if and only if A is a yes-instance of EXACT PINWHEEL PACKING.*

Proof. We give a reduction from 3-COLORING IN GRAPHS WITH MAXIMUM DEGREE FOUR to EXACT PINWHEEL PACKING. It suffices to consider the case where n is sufficiently large.

Reduction. Let $G = (V, E)$ be a graph with $|V| = n$ whose maximum degree is four. We first construct a set T and a family $(T_v)_{v \in V}$ of subsets of T by applying the following technical lemma, whose proof is postponed to Section 2.3.

► **Lemma 7.** *Let $G = (V, E)$ be a graph with $|V| = n$ whose maximum degree is four. In $n^{O(1)}$ time, we can construct a set T with $|T| = O(\log n)$ and its subsets $T_v \subseteq T$ each indexed by $v \in V$ such that vertices v and v' are adjacent in G if and only if $T_v \cap T_{v'} = \emptyset$.*

We pick $|T|$ distinct primes $(p_i)_{i \in T}$ indexed by T such that $n < p_i < 2n$ for each $i \in T$. Note that such primes exist when n is sufficiently large, because the prime number theorem guarantees that, for any positive integer N , there are $\Theta\left(\frac{N}{\log N}\right)$ primes less than or equal to N . For every $v \in V$, let

$$a_v = 3 \cdot \prod_{i \in T_v} p_i.$$

Then, we obtain an instance $A = (a_v)_{v \in V}$ of EXACT PINWHEEL PACKING. The construction can be done in $n^{O(1)}$ -time, and

$$\text{lcm}(A) \leq 3 \cdot \left(\max_{i \in T} p_i \right)^{|T|} = n^{O(\log n)}.$$

Completeness. We show that if G is 3-colorable, then $A = (a_v)_{v \in V}$ is a yes-instance of EXACT PINWHEEL PACKING. Suppose $G = (V, E)$ has a 3-coloring $c: V \rightarrow [3]$. Let ℓ be a bijection from V to $[n]$. For each $v \in V$, let r_v be an integer such that

$$r_v \equiv c(v) \pmod{3}, \tag{2}$$

and

$$r_v \equiv \ell(v) \pmod{\prod_{i \in T_v} p_i}, \tag{3}$$

whose existence is guaranteed by the Chinese remainder theorem. Let $S_v = \{r_v + a_v \cdot q : q \in \mathbb{Z}\}$ for $v \in V$. We claim that $(S_v)_{v \in V}$ is a solution to the EXACT PINWHEEL PACKING instance $A = (a_v)_{v \in V}$, that is, $S_v \cap S_{v'} = \emptyset$ holds for any pair of distinct $v, v' \in V$. We consider the following two cases separately.

- Suppose that v and v' are adjacent in G . Then, we have $c(v) \neq c(v')$ as c is a coloring, and hence $r_v \not\equiv r_{v'} \pmod{3}$ by (2). Thus, for all $q, q' \in \mathbb{Z}$,

$$r_v + a_v \cdot q \neq r_{v'} + a_{v'} \cdot q',$$

because both a_v and $a_{v'}$ are divisible by 3. This shows that $S_v \cap S_{v'} = \emptyset$.

- Suppose that v and v' are not adjacent in G . Then, $T_v \cap T_{v'} \neq \emptyset$ holds by the construction in Lemma 7. Let i be an element in $T_v \cap T_{v'}$. Since $p_i > n > |\ell(v) - \ell(v')| \geq 1$, we have $\ell(v) \not\equiv \ell(v') \pmod{p_i}$, which implies that

$$r_v \equiv \ell(v) \not\equiv \ell(v') \equiv r_{v'} \pmod{p_i}$$

by (3). Thus, for all $q, q' \in \mathbb{Z}$,

$$r_v + a_v \cdot q \neq r_{v'} + a_{v'} \cdot q',$$

because both a_v and $a_{v'}$ are divisible by p_i . This shows that $S_v \cap S_{v'} = \emptyset$.

Soundness. We show that if $A = (a_v)_{v \in V}$ is a yes-instance of EXACT PINWHEEL PACKING, then G is 3-colorable. Suppose that $(S_v)_{v \in V}$ is a solution to the EXACT PINWHEEL PACKING instance $A = (a_v)_{v \in V}$. For each $v \in V$, since S_v is a residue class modulo a_v , there exists an integer r_v such that $S_v = \{r_v + a_v \cdot q : q \in \mathbb{Z}\}$. For $v \in V$, let $c(v)$ be the integer in $\{1, 2, 3\}$ such that $c(v) \equiv r_v \pmod{3}$. This defines a function $c: V \rightarrow [3]$. We claim that c is a coloring of G . Let $v, v' \in V$ be vertices that are adjacent to each other. Then, we have $T_v \cap T_{v'} = \emptyset$, and hence $\gcd(a_v, a_{v'}) = 3$, where $\gcd$ denotes the greatest common divisor. Since $S_v \cap S_{v'} = \emptyset$ is equivalent to $r_v \not\equiv r_{v'} \pmod{\gcd(a_v, a_{v'})}$ by the Chinese remainder theorem (see e.g. [3, Lemma 12]), we obtain

$$c(v) \equiv r_v \not\equiv r_{v'} \equiv c(v') \pmod{3}.$$

Therefore, c is a 3-coloring of G . ◀

► **Remark 8.** Theorem 6 gives an alternative proof of the NP-completeness of EXACT PINWHEEL PACKING, which has already been shown in [3, 16, 21, 25]. The condition $\text{lcm}(A) = n^{O(\log n)}$ is not required to prove the NP-completeness of EXACT PINWHEEL PACKING. However, it is used to show the hardness of DENSE PINWHEEL PACKING in the next subsection.

2.2 Hardness of Dense Pinwheel Packing

In this subsection, we prove the hardness of DENSE PINWHEEL PACKING using Theorem 6.

► **Theorem 9.** *Let $G = (V, E)$ be a graph with $|V| = n$ whose maximum degree is four. In $n^{O(\log n)}$ time, we can construct an instance $\hat{A} = (a_i)_{i \in [k]}$ of DENSE PINWHEEL PACKING such that $k = n^{O(\log n)}$, $a_i = n^{O(\log n)}$ for each $i \in [k]$, and $G = (V, E)$ is 3-colorable if and only if $\hat{A}$ is a yes-instance of DENSE PINWHEEL PACKING.*

Proof. Let $G = (V, E)$ be a graph with $|V| = n$ whose maximum degree is four. By Theorem 6, in $n^{O(1)}$ time, we construct a sequence of positive integers $A = (a_i)_{i \in [\ell]}$ with $\text{lcm}(A) = n^{O(\log n)}$ such that $G = (V, E)$ is 3-colorable if and only if A is a yes-instance of EXACT PINWHEEL PACKING. We may assume that the density of A is at most 1, since otherwise we can just return a trivial no-instance.

We construct an instance $\hat{A}$ of DENSE PINWHEEL PACKING from A by appending multiple copies of integer $\text{lcm}(A)$ so that the density becomes 1. That is, we define $\hat{A} = (a_i)_{i \in [k]}$ by

$$k = \ell + \left(1 - \sum_{i=1}^{\ell} \frac{1}{a_i}\right) \text{lcm}(A),$$

$$a_{\ell+1} = a_{\ell+2} = \dots = a_k = \text{lcm}(A).$$

Then, we see that $k \leq \text{lcm}(A) = n^{O(\log n)}$ and $a_i \leq \text{lcm}(A) = n^{O(\log n)}$ for each $i \in [k]$. This implies that $\hat{A}$ can be constructed in $n^{O(\log n)}$ time.

It remains to show that $G = (V, E)$ is 3-colorable if and only if $\hat{A}$ is a yes-instance of DENSE PINWHEEL PACKING. To this end, we prove that A is a yes-instance of EXACT PINWHEEL PACKING if and only if $\hat{A}$ is a yes-instance of DENSE PINWHEEL PACKING.

Suppose first that $\hat{A}$ is a yes-instance of DENSE PINWHEEL PACKING and let $(S_i)_{i \in [k]}$ be a solution for it. Then, $(S_i)_{i \in [\ell]}$ is a solution to the EXACT PINWHEEL PACKING instance A .

Suppose next that A is a yes-instance of EXACT PINWHEEL PACKING and let $(S_i)_{i \in [\ell]}$ be a solution for it. Since each S_i is a residue class modulo a_i , it is the union of $\frac{\text{lcm}(A)}{a_i}$ residue classes modulo $\text{lcm}(A)$. Then, $\mathbb{Z} \setminus \bigcup_{i=1}^{\ell} S_i$ is the union of $\text{lcm}(A) - \sum_{i=1}^{\ell} \frac{\text{lcm}(A)}{a_i} = k - \ell$ residue classes modulo $\text{lcm}(A)$. Therefore, we can take residue classes $S_{\ell+1}, \dots, S_k$ modulo $\text{lcm}(A)$ so that $(S_i)_{i \in [k]}$ is a solution to the DENSE PINWHEEL PACKING instance $\hat{A}$. ◀

We are now ready to prove Theorem 2, which we restate here.

► **Theorem 2.** DENSE PINWHEEL PACKING *cannot be solved in polynomial time unless* $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$.

Proof. We prove $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$ assuming that DENSE PINWHEEL PACKING can be solved in polynomial time. Suppose we are given an instance of a problem in NP whose input size is n . Then, we can construct an equivalent instance of 3-COLORING IN GRAPHS WITH MAXIMUM DEGREE FOUR whose size is $n^{O(1)}$, because 3-COLORING IN GRAPHS WITH MAXIMUM DEGREE FOUR is NP-hard (see [8]). By Theorem 9, we construct an equivalent instance of DENSE PINWHEEL PACKING whose size is $n^{O(\log n)}$. These reductions show that, if DENSE PINWHEEL PACKING can be solved in polynomial time, then the original problem in NP can be solved in $n^{O(\log n)}$ time. Therefore, $\text{NP} \subseteq \text{DTIME}(n^{O(\log n)})$ holds under the assumption that DENSE PINWHEEL PACKING can be solved in polynomial time. ◀

Using the same reduction, we can also show that DENSE PINWHEEL PACKING cannot be solved in quasi-polynomial time (i.e., $2^{\text{polylog}(n)}$ time) under a stronger complexity assumption. Note that the same hardness result holds also for PINWHEEL PACKING and PINWHEEL COVERING, because DENSE PINWHEEL PACKING is a special case of these problems.

► **Corollary 10.** DENSE PINWHEEL PACKING *cannot be solved in quasi-polynomial time unless* $\text{NP} \subseteq \text{QP}$, *where QP is the class of problems that can be solved in quasi-polynomial time, i.e.,* $\text{QP} = \text{DTIME}(2^{\text{polylog}(n)})$.

Proof. As in the proof of Theorem 2, any problem in NP with input size n can be reduced to DENSE PINWHEEL PACKING with size $N = n^{O(\log n)}$. If DENSE PINWHEEL PACKING can be solved in quasi-polynomial time, then the original problem can be solved in $2^{\text{polylog}(N)} = 2^{\text{polylog}(n)}$ time. Therefore, $\text{NP} \subseteq \text{QP}$ holds under the assumption that DENSE PINWHEEL PACKING can be solved in quasi-polynomial time. ◀

2.3 Proof of Lemma 7

In this subsection, we give a proof of Lemma 7. Let $G = (V, E)$ be a graph with $|V| = n$ whose maximum degree is four. A vertex subset $I \subseteq V$ is called an *independent set* if no two vertices in I are adjacent to each other. Our idea is to find a family $\mathcal{I}$ of independent sets in G such that

(I) for any pair of non-adjacent vertices $v, v' \in V$, there exists $I \in \mathcal{I}$ such that $\{v, v'\} \subseteq I$.

► **Claim 11.** Suppose we are given an independent set family $\mathcal{I}$ of size $O(\log n)$ with property (I). Then, in polynomial time, we can construct a set T and a subset $T_v \subseteq T$ for each $v \in V$ satisfying the conditions in Lemma 7.

Proof of Claim 11. Let $\mathcal{I}$ be an independent set family of size $O(\log n)$ with property (I). Then, we can simply set $T = \mathcal{I}$ and let $T_v = \{I \in \mathcal{I} : v \in I\}$ for $v \in V$. We easily see that such T and $(T_v)_{v \in V}$ satisfy the conditions in Lemma 7 as follows.

- It is obvious that $|T| = |\mathcal{I}| = O(\log n)$.
- If vertices v and v' are adjacent, then they are not contained in the same independent set in G , and hence $T_v \cap T_{v'} = \emptyset$.
- If vertices v and v' are not adjacent, then there must exist an independent set $I \in \mathcal{I}$ such that $\{v, v'\} \subseteq I$ by property (I), and hence $I \in T_v \cap T_{v'}$.

This completes the proof of the claim. ◀

To find an independent set family $\mathcal{I}$ of size $O(\log n)$ that satisfies property (I), we use an argument inspired by the color-coding technique [1]. For every $v \in V$, let $N[v]$ denote its closed neighborhood, i.e., $N[v] = \{u \in V : vu \in E\} \cup \{v\}$. Observe that $|N[v] \cup N[v']| \leq 10$ for any distinct vertices $v, v' \in V$ as G is a graph with maximum degree four. We show that it suffices to construct a family $\mathcal{U} \subseteq 2^V$ of vertex subsets with $|\mathcal{U}| = O(\log n)$ such that

(U) for any pair of distinct vertices $v, v' \in V$, there exists a vertex subset $U \in \mathcal{U}$ such that $U \cap (N[v] \cup N[v']) = \{v, v'\}$.

▷ **Claim 12.** Suppose we are given a family $\mathcal{U} \subseteq 2^V$ of vertex subsets that satisfies property (U). Then, in polynomial time, we can construct an independent set family $\mathcal{I}$ of size at most $|\mathcal{U}|$ with property (I).

Proof of Claim 12. For each $U \in \mathcal{U}$, let I_U be an inclusionwise maximal independent set in G subject to $I_U \subseteq U$. Then, set $\mathcal{I} = \{I_U : U \in \mathcal{U}\}$, which can be computed in polynomial time. It is obvious that $|\mathcal{I}| \leq |\mathcal{U}|$.

Let $v, v' \in V$ be a pair of distinct non-adjacent vertices. By property (U), there exists a vertex subset $U \in \mathcal{U}$ such that $U \cap (N[v] \cup N[v']) = \{v, v'\}$. Since any maximal independent set contained in U contains both v and v' , we obtain $\{v, v'\} \subseteq I_U$. This shows that $\mathcal{I}$ satisfies property (I). ◁

By Claims 11 and 12, to prove Lemma 7, it remains to construct in polynomial time a family $\mathcal{U} \subseteq 2^V$ with $|\mathcal{U}| = O(\log n)$ that satisfies property (U). Below, we describe a simple probabilistic construction of $\mathcal{U}$, followed by a deterministic construction using a sophisticated tool known as the (n, k) -universal sets.

Probabilistic Construction of $\mathcal{U}$

Consider a random vertex subset $U \subseteq V$ that contains each vertex independently with probability $\frac{1}{2}$. Then, for any pair of distinct vertices $v, v' \in V$, $U \cap (N[v] \cup N[v']) = \{v, v'\}$ holds with probability

$$\frac{1}{2^{|N[v] \cup N[v']|}} \geq \frac{1}{2^{10}}.$$

We pick such a random vertex subset U independently many times to obtain a family $\mathcal{U}$. Then, for a pair of distinct vertices $v, v' \in V$, $U \cap (N[v] \cup N[v']) \neq \{v, v'\}$ holds for every $U \in \mathcal{U}$ with probability at most

$$\left(1 - \frac{1}{2^{10}}\right)^{|\mathcal{U}|}.$$

By the union bound, with probability at least $1 - |V|^2 \left(1 - \frac{1}{2^{10}}\right)^{|\mathcal{U}|}$, $\mathcal{U}$ satisfies property (U). When $|\mathcal{U}| = \Theta(\log |V|) = \Theta(\log n)$, we can generate such a family $\mathcal{U}$ with constant probability in polynomial time.

Deterministic Construction of $\mathcal{U}$

We use the (n, k) -universal sets to construct a desired family $\mathcal{U}$ deterministically.

► **Definition 13** ((n, k) -universal set). For positive integers n and k , an (n, k) -universal set is a set of vectors $W \subseteq \{0, 1\}^n$ such that for any index set $S \subseteq [n]$ of size k , the projection of W on S contains all 2^k possible configurations.

► **Theorem 14** ([26, Theorem 2]). *For any positive integers n and k with $k \leq n$, an (n, k) -universal set of cardinality $O(k2^k \log n)$ can be constructed deterministically in time $O(\binom{n}{k} k 2^{2k} n^{\lfloor k/2 \rfloor})$.*

Setting $n = |V|$ and $k = 10$, by Theorem 14, we construct an (n, k) -universal set W of size $O(k2^k \log n) = O(\log n)$ in time $O(\binom{n}{k} k 2^{2k} n^{\lfloor k/2 \rfloor}) = n^{O(1)}$. Using an arbitrary bijection from V to $[n]$, $W \subseteq \{0, 1\}^n$ can be regarded as a subset of $\{0, 1\}^V$. By identifying a subset of V and its characteristic vector in $\{0, 1\}^V$, W defines a family $\mathcal{U} \subseteq 2^V$. Then, $|\mathcal{U}| = |W| = O(\log n)$.

We see that $\mathcal{U}$ satisfies property (U) as follows. Let $v, v' \in V$ be distinct non-adjacent vertices. Since W is a $(|V|, 10)$ -universal set and $|N[v] \cup N[v']| \leq 10$, there exists a vector $w \in W$ whose projection on $N[v] \cup N[v']$ is the characteristic vector of $\{v, v'\}$. This means that the set $U \in \mathcal{U}$ corresponding to w satisfies $U \cap (N[v] \cup N[v']) = \{v, v'\}$.

3 Fixed Parameter Algorithms

In this section, we present fixed parameter algorithms for (EXACT) PINWHEEL COVERING and EXACT PINWHEEL PACKING. We prove Theorems 4 and 5 in Sections 3.1 and 3.2, respectively.

3.1 Pinwheel Covering and Exact Pinwheel Covering

In this subsection, we deal with PINWHEEL COVERING and EXACT PINWHEEL COVERING simultaneously and prove Theorem 4. We first present the following claim. Although this claim was already shown in [21, Section 4.1] for PINWHEEL COVERING, we here give a proof for completeness.

► **Claim 15** ([21, Section 4.1]). PINWHEEL COVERING and EXACT PINWHEEL COVERING can be solved in $\left(\prod_{i=1}^k a_i\right)^{O(1)}$ time.

Proof of Claim 15. We first consider PINWHEEL COVERING. Construct a digraph G with $\prod_{i=1}^k a_i$ vertices, each of which corresponds to a k -tuple $(b_1, \dots, b_k)$ with $b_i \in \{0, 1, \dots, a_i - 1\}$ for each $i \in [k]$. This vertex corresponds to a state in which agent i must wait for b_i days before handling the task again. We add an arc from a vertex $(b_1, \dots, b_k)$ to a vertex $(b'_1, \dots, b'_k)$ when there is $i \in [k]$ such that $b_i = 0$, $b'_i = a_i - 1$, and $b'_j = \max\{0, b_j - 1\}$ for $j \in [k] \setminus \{i\}$. Since a feasible solution of PINWHEEL COVERING corresponds to an infinite walk in G , A is a yes-instance if and only if G contains a dicycle. This can be checked in $\left(\prod_{i=1}^k a_i\right)^{O(1)}$ time.

To deal with EXACT PINWHEEL COVERING, we modify the definition of the edge set as follows: we add an arc from a vertex $(b_1, \dots, b_k)$ to a vertex $(b'_1, \dots, b'_k)$ when there is $i \in [k]$ such that $b_i = 0$, $b'_i = a_i - 1$, and $(b_j, b'_j) = (0, a_j - 1)$ or $b'_j = b_j - 1$ for $j \in [k] \setminus \{i\}$.

Then, A is a yes-instance if and only if the digraph contains a dicycle. This can be checked in $\left(\prod_{i=1}^k a_i\right)^{O(1)}$ time. ◁

By this claim, to obtain an FPT algorithm for (EXACT) PINWHEEL COVERING, it is sufficient to bound each a_i by a function of k . To achieve this, we prove the following two claims.

► **Claim 16.** Suppose that $A = (a_i)_{i \in [k]}$ is a no-instance of (EXACT) PINWHEEL COVERING and $(S_i)_{i \in [k]}$ is a family of subsets of $\mathbb{Z}$ such that

$$|[m, m + a_i) \cap S_i| \leq 1 \quad (\text{or } |[m, m + a_i) \cap S_i| = 1)$$

for all $i \in [k]$ and $m \in \mathbb{Z}$. Then, no $\prod_{i=1}^k a_i$ consecutive integers are covered by $\bigcup_{i=1}^k S_i$, that is,

$$\left| \left[m, m + \prod_{i=1}^k a_i \right) \cap \left(\mathbb{Z} - \bigcup_{i=1}^k S_i \right) \right| \geq 1 \quad (4)$$

for all $m \in \mathbb{Z}$.

Proof of Claim 16. Suppose that $A = (a_i)_{i \in [k]}$ is a no-instance of (EXACT) PINWHEEL COVERING. Then, the digraph in the proof of Claim 15 contains no dicycle. Therefore, the digraph contains no walk of length $\prod_{i=1}^k a_i$, which means that no $\prod_{i=1}^k a_i$ consecutive integers are covered by $\bigcup_{i=1}^k S_i$. $\triangleleft$

$\triangleright$ **Claim 17.** Let $A = (a_i)_{i \in [k]}$ be a sequence of positive integers with $a_1 \leq a_2 \leq \dots \leq a_k$, and let $\ell \in [k-1]$. If A is a yes-instance of (EXACT) PINWHEEL COVERING and $A' = (a_i)_{i \in [\ell]}$ is a no-instance of (EXACT) PINWHEEL COVERING, then $a_{\ell+1} \leq (k-\ell) \prod_{i=1}^{\ell} a_i$.

Proof of Claim 17. Let $(S_i)_{i \in [k]}$ be a feasible solution for the instance A . Since $A' = (a_i)_{i \in [\ell]}$ is a no-instance of (EXACT) PINWHEEL COVERING, by Claim 16, no $\prod_{i=1}^{\ell} a_i$ consecutive integers are covered by $\bigcup_{i=1}^{\ell} S_i$. Therefore, in any $\prod_{i=1}^{\ell} a_i$ consecutive integers, at least one integer is covered by $S_{\ell+1} \cup \dots \cup S_k$. This implies that

$$\frac{1}{\prod_{i=1}^{\ell} a_i} \leq \frac{1}{a_{\ell+1}} + \dots + \frac{1}{a_k} \leq \frac{k-\ell}{a_{\ell+1}},$$

and hence $a_{\ell+1} \leq (k-\ell) \prod_{i=1}^{\ell} a_i$. $\triangleleft$

We are now ready to prove Theorem 4, which we restate here.

► Theorem 4. PINWHEEL COVERING and EXACT PINWHEEL COVERING can be solved in $O(n) + k^{O(2^k)}$ time, where $n := \sum_{i=1}^k \log a_i$ is the input size of the problem.

Proof. Let $A = (a_i)_{i \in [k]}$ be an instance of (EXACT) PINWHEEL COVERING. Without loss of generality, we may assume that $a_1 \leq a_2 \leq \dots \leq a_k$. It suffices to consider the case where $a_1 \leq k$, since otherwise we can immediately conclude that A is a no-instance.

If $a_i \leq k^{2^{i-1}}$ for every $i \in [k]$, then we can check the feasibility of A in $\left(\prod_{i=1}^k a_i\right)^{O(1)} = k^{O(2^k)}$ time by Claim 15. Otherwise, let $\ell \in [k-1]$ be the minimum integer with $a_{\ell+1} > k^{2^{\ell}}$, which can be computed in $O(n)$ time. Then, we obtain

$$a_{\ell+1} > k^{2^{\ell}} = k \prod_{i=1}^{\ell} k^{2^{i-1}} > (k-\ell) \prod_{i=1}^{\ell} k^{2^{i-1}} > (k-\ell) \prod_{i=1}^{\ell} a_i, \quad (5)$$

where the last inequality is by the minimality of ℓ and $a_1 < k^{2^{1-1}}$. We observe that $(a_i)_{i \in [k]}$ is a yes-instance if and only if $(a_i)_{i \in [\ell]}$ is a yes-instance, as the “if” part is trivial and the “only if” part follows from Claim 17 and (5). We can check whether $(a_i)_{i \in [\ell]}$ is a yes-instance or not in $\left(\prod_{i=1}^{\ell} a_i\right)^{O(1)}$ time by Claim 15, and this running time is bounded by $k^{O(2^k)}$. $\blacktriangleleft$

3.2 Exact Pinwheel Packing

In this subsection, we deal with EXACT PINWHEEL PACKING and prove Theorem 5.

► **Theorem 5.** EXACT PINWHEEL PACKING can be solved in $(\log k)^{O(k^2)} \cdot n^{O(1)}$ time, where $n := \sum_{i=1}^k \log a_i$ is the input size of the problem.

Proof. Let $A = (a_i)_{i \in [k]}$ be an instance of EXACT PINWHEEL PACKING. Recall that the objective is to find pairwise disjoint sets $S_1, \dots, S_k$ such that S_i is a residue class modulo a_i for $i \in [k]$. This is equivalent to finding $r_i \in \mathbb{Z}$ for $i \in [k]$ such that

$$S_i = \{r_i + a_i \cdot q : q \in \mathbb{Z}\} \text{ and } S_j = \{r_j + a_j \cdot q : q \in \mathbb{Z}\} \text{ are disjoint} \quad (6)$$

for all distinct $i, j \in [k]$. By the Chinese remainder theorem (see e.g. [3, Lemma 12]), (6) is equivalent to

$$r_i \not\equiv r_j \pmod{\gcd(a_i, a_j)}. \quad (7)$$

We now reduce this problem to the integer linear programming problem. We treat $r_1, \dots, r_k$ as integer variables. In addition to these, for all distinct $i, j \in [k]$, we introduce new integer variables $q_{i,j}$ and $\ell_{i,j}$, and consider the following linear constraints:

$$r_i - r_j = q_{i,j} \cdot \gcd(a_i, a_j) + \ell_{i,j}, \quad (8)$$

$$1 \leq \ell_{i,j} \leq \gcd(a_i, a_j) - 1. \quad (9)$$

Then, we obtain an integer linear program with $O(k^2)$ variables and $O(k^2)$ constraints. Since r_i and r_j satisfy (7) if and only if (8) and (9) hold for some integers $q_{i,j}$ and $\ell_{i,j}$, EXACT PINWHEEL PACKING is reduced to the integer linear program defined by (8) and (9).

It is known that the integer linear programming problem can be solved in FPT time, parameterized by its variable number p ; see e.g., [17, 23, 27], and the current smallest parameter dependency is $(\log p)^{O(p)}$ [27]. Using this result, we obtain a $(\log k)^{O(k^2)} \cdot n^{O(1)}$ time algorithm for EXACT PINWHEEL PACKING. ◀

References

- 1 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, July 1995. doi:10.1145/210332.210337.
- 2 Paul Balister. *Erdős Covering Systems*, pages 31–54. London Mathematical Society Lecture Note Series. Cambridge University Press, 2024.
- 3 Amotz Bar-Noy, Randeep Bhatia, Joseph Naor, and Baruch Schieber. Minimizing service and operation costs of periodic scheduling. *Mathematics of Operations Research*, 27(3):518–544, 2002. doi:10.1287/moor.27.3.518.314.
- 4 Amotz Bar-Noy and Richard E. Ladner. Windows scheduling problems for broadcast systems. *SIAM Journal on Computing*, 32(4):1091–1113, 2003. doi:10.1137/S009753970240447X.
- 5 Amotz Bar-Noy, Richard E. Ladner, and Tami Tamir. Windows scheduling as a restricted version of bin packing. *ACM Trans. Algorithms*, 3(3):Article 28, 2007. doi:10.1145/1273340.1273344.
- 6 Mee Yee Chan and Francis Chin. Schedulers for larger classes of pinwheel instances. *Algorithmica*, 9:425–462, 1993. doi:10.1007/BF01187034.
- 7 Peter C. Fishburn and J. C. Lagarias. Pinwheel scheduling: Achievable densities. *Algorithmica*, 34(1):14–38, 2002. doi:10.1007/S00453-002-0938-9.
- 8 M. R. Garey and D. S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W.H. Freeman and Co, 2003.

- 9 Leszek Gąsieniec, Ralf Klasing, Christos Levcopoulos, Andrzej Lingas, Jie Min, and Tomasz Radzik. Bamboo garden trimming problem (perpetual maintenance of machines with different attendance urgency factors). In Bernhard Steffen, Christel Baier, Mark van den Brand, Johann Eder, Mike Hinchey, and Tiziana Margaria, editors, *SOFSEM 2017: Theory and Practice of Computer Science*, pages 229–240, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-51963-0_18.
- 10 Leszek Gąsieniec, Benjamin Smith, and Sebastian Wild. Towards the 5/6-density conjecture of pinwheel scheduling. In Cynthia A. Phillips and Bettina Speckmann, editors, *Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX 2022, Alexandria, VA, USA, January 9-10, 2022*, pages 91–103. SIAM, 2022. doi:10.1137/1.9781611977042.8.
- 11 Leszek Gąsieniec, Tomasz Jurdziński, Ralf Klasing, Christos Levcopoulos, Andrzej Lingas, Jie Min, and Tomasz Radzik. Perpetual maintenance of machines with different urgency requirements. *Journal of Computer and System Sciences*, 139:103476, 2024. doi:10.1016/j.jcss.2023.103476.
- 12 Robert Holte, Aloysius Mok, Louis Rosier, Igor Tulchinsky, and Donald Varvel. The pinwheel: a real-time scheduling problem. In *Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences. Volume II: Software Track*, volume 2, pages 693–702 vol.2, 1989. doi:10.1109/HICSS.1989.48075.
- 13 Robert Holte, Louis Rosier, Igor Tulchinsky, and Donald Varvel. Pinwheel scheduling with two distinct numbers. *Theoretical Computer Science*, 100(1):105–135, 1992. doi:10.1016/0304-3975(92)90365-M.
- 14 Bob Hough. Solution of the minimum modulus problem for covering systems. *Annals of Mathematics*, 181(1):361–382, 2015. doi:10.4007/annals.2015.181.1.6.
- 15 A.P. Huhn and L. Megyesi. On disjoint residue classes. *Discrete Mathematics*, 41(3):327–330, 1982. doi:10.1016/0012-365X(82)90030-9.
- 16 Tobias Jacobs and Salvatore Longo. A new perspective on the windows scheduling problem, 2014. arXiv:1410.7237.
- 17 Ravi Kannan. Minkowski’s convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987. doi:10.1287/moor.12.3.415.
- 18 Akitoshi Kawamura. Proof of the density threshold conjecture for pinwheel scheduling. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, pages 1816–1819, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3618260.3649757.
- 19 Akitoshi Kawamura. Perpetual scheduling under frequency constraints. In Shin-ichi Minato, Takeaki Uno, Norihito Yasuda, Takashi Horiyama, Ken-ichi Kawarabayashi, Shigeru Yamashita, and Hirotaka Ono, editors, *Algorithmic Foundations for Social Advancement: Recent Progress on Theory and Practice*. Springer, 2025. doi:10.1007/978-981-96-0668-9.
- 20 Akitoshi Kawamura, Yosuke Kusano, and Yusuke Kobayashi. Pinwheel covering. In *The 14th International Conference on Algorithms and Complexity (CIAC)*, pages 185–199, 2025. doi:10.1007/978-3-031-92935-9_12.
- 21 Akitoshi Kawamura and Makoto Soejima. Simple strategies versus optimal schedules in multi-agent patrolling. *Theoretical Computer Science*, 839:195–206, 2020. doi:10.1016/j.tcs.2020.07.037.
- 22 Jan Korst, Emile Aarts, and Jan Karel Lenstra. Scheduling periodic tasks. *INFORMS Journal on Computing*, 8(4):428–435, 1996. doi:10.1287/ijoc.8.4.428.
- 23 Hendrik W Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of operations research*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538.
- 24 Shun-Shii Lin and Kwei-Jay Lin. A pinwheel scheduler for three distinct numbers with a tight schedulability bound. *Algorithmica*, 19:411–426, 1997. doi:10.1007/PL00009181.
- 25 Al Mok, Louis Rosier, Igor Tulchinsky, and Donald Varvel. Algorithms and complexity of the periodic maintenance problem. *Microprocessing and Microprogramming*, 27(1):657–664, 1989. Fifteenth EUROMICRO Symposium on Microprocessing and Microprogramming. doi:10.1016/0165-6074(89)90128-2.

- 26 Moni Naor, Leonard J Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 182–191. IEEE, 1995. doi:10.1109/SFCS.1995.492475.
- 27 Victor Reis and Thomas Rothvoss. The subspace flatness conjecture and faster integer programming. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 974–988, 2023. doi:10.1109/FOCS57990.2023.00060.
- 28 Zhi-Wei Sun. On disjoint residue classes. *Discrete Mathematics*, 104(3):321–326, 1992. doi:10.1016/0012-365X(92)90454-N.
- 29 Martijn van Ee. A 12/7-approximation algorithm for the discrete bamboo garden trimming problem. *Operations Research Letters*, 49(5):645–649, 2021. doi:10.1016/j.orl.2021.07.001.
- 30 W.D Wei and C.L Liu. On a periodic maintenance problem. *Operations Research Letters*, 2(2):90–93, 1983. doi:10.1016/0167-6377(83)90044-5.

A An FPT Algorithm for Pinwheel Packing

As described in Section 1.3, an FPT algorithm for PINWHEEL PACKING was already presented in [10] (without an explicit running time bound). In this section, to highlight the similarities and differences between PINWHEEL PACKING and (EXACT) PINWHEEL COVERING, building on the argument in [10], we present an FPT algorithm for PINWHEEL PACKING that runs in $O(n) + k^{O(2^k)}$ time.

An instance $(a_i)_{i \in [k]}$ of PINWHEEL PACKING is called a *strong yes-instance* if there exist pairwise disjoint sets $S_1, \dots, S_k \subseteq \mathbb{Z}$ such that $\mathbb{Z} - \bigcup_{i=1}^k S_i$ contains an infinite number of integers. We first present the following claim that was shown in [10] with a different terminology.

▷ **Claim 18** (see [10, Proposition 2.1]). PINWHEEL PACKING can be solved in $\left(\prod_{i=1}^k a_i\right)^{O(1)}$ time. We can also check whether a given instance $(a_i)_{i \in [k]}$ is a strong yes-instance in $\left(\prod_{i=1}^k a_i\right)^{O(1)}$ time.

Proof of Claim 18. We construct a digraph G with $\prod_{i=1}^k a_i$ vertices, each of which corresponds to a k -tuple $(b_1, \dots, b_k)$ with $b_i \in \{0, 1, \dots, a_i - 1\}$ for each $i \in [k]$. This vertex corresponds to a state where task i must be performed within b_i days from the present. We define two types of arcs in G , which we call *normal arcs* and *lazy arcs*. We add a normal arc from a vertex $(b_1, \dots, b_k)$ to a vertex $(b'_1, \dots, b'_k)$ when there is $i \in [k]$ such that $b'_i = a_i - 1$, and $b'_j = b_j - 1$ for $j \in [k] \setminus \{i\}$. We add a lazy arc from a vertex $(b_1, \dots, b_k)$ to a vertex $(b'_1, \dots, b'_k)$ when $b'_j = b_j - 1$ for $j \in [k]$. Note that traversing a normal arc corresponds to performing task i , while traversing a lazy arc corresponds to performing no task. Since a feasible solution of PINWHEEL COVERING corresponds to an infinite walk in G , A is a yes-instance if and only if G contains a dicycle. Similarly, A is a strong yes-instance if and only if G contains a dicycle containing a lazy arc. Both conditions can be checked in $\left(\prod_{i=1}^k a_i\right)^{O(1)}$ time using depth first search. ◁

In a similar way to (EXACT) PINWHEEL COVERING, we prove the following two claims to bound each a_i by a function of k .

▷ **Claim 19.** Suppose that $A = (a_i)_{i \in [k]}$ is a strong yes-instance of PINWHEEL PACKING. Then, there exists a feasible solution $(S_i)_{i \in [k]}$ such that at least one integer is not covered by $\bigcup_{i=1}^k S_i$ in every consecutive $\prod_{i=1}^k a_i$ integers, that is, (4) holds for all $m \in \mathbb{Z}$.

Proof of Claim 19. If A is a strong yes-instance, then the digraph in the proof of Claim 18 has a dicycle containing a lazy arc. Let $(S_i)_{i \in [k]}$ be a feasible solution of PINWHEEL PACKING that corresponds to an infinite walk traversing such a dicycle repeatedly. Since the length of the dicycle is at most $\prod_{i=1}^k a_i$, in every consecutive $\prod_{i=1}^k a_i$ steps along this walk, we traverse at least one lazy arc. This means that, in every consecutive $\prod_{i=1}^k a_i$ days, there is at least one day on which no task is performed. Therefore, (4) holds for all $m \in \mathbb{Z}$. $\triangleleft$

$\triangleright$ **Claim 20.** Let $A = (a_i)_{i \in [k]}$ be a sequence of positive integers with $a_1 \leq a_2 \leq \dots \leq a_k$, and let $\ell \in [k-1]$. If $A' = (a_i)_{i \in [\ell]}$ is a strong yes-instance of PINWHEEL PACKING and $a_{\ell+1} \geq (k-\ell) \prod_{i=1}^{\ell} a_i$, then A is a yes-instance of PINWHEEL PACKING.

Proof of Claim 20. Suppose that $A' = (a_i)_{i \in [\ell]}$ is a strong yes-instance of PINWHEEL PACKING and $a_{\ell+1} \geq (k-\ell) \prod_{i=1}^{\ell} a_i$. By Claim 19, there exist pairwise disjoint sets $S_1, \dots, S_{\ell} \subseteq \mathbb{Z}$ such that at least one integer is not covered by $\bigcup_{i=1}^{\ell} S_i$ in every consecutive $\prod_{i=1}^{\ell} a_i$ integers. Let $(x_j)_{j \in \mathbb{Z}}$ be an infinite sequence of all integers not covered by $\bigcup_{i=1}^{\ell} S_i$, which are arranged in ascending order. For $h \in \{\ell+1, \ell+2, \dots, k\}$, set

$$S_h = \{x_j : j \equiv h \pmod{k-\ell}\}.$$

Then, obviously, $S_1, \dots, S_k$ are pairwise disjoint. For $h \in \{\ell+1, \ell+2, \dots, k\}$, since

$$x_{j+k-\ell} - x_j \leq (k-\ell) \prod_{i=1}^{\ell} a_i \leq a_{\ell+1} \leq a_h$$

holds for every $j \in \mathbb{Z}$, we see that task h is performed every a_h days. Therefore, $(S_i)_{i \in [k]}$ is a feasible solution for the instance A . $\triangleleft$

We are now ready to show the following theorem.

Theorem 21 (see [10, Corollary 3.2]). PINWHEEL PACKING can be solved in $O(n) + k^{O(2^k)}$ time, where $n := \sum_{i=1}^k \log a_i$ is the input size of the problem.

Proof. Let $A = (a_i)_{i \in [k]}$ be an instance of PINWHEEL PACKING. Without loss of generality, we may assume that $a_1 \leq a_2 \leq \dots \leq a_k$. If $a_1 \geq k$, then A is a yes-instance, as we can simply perform each task exactly once every k days. Thus, it suffices to consider the case where $a_1 < k$.

If $a_i \leq k^{2^{i-1}}$ for every $i \in [k]$, then we can check the feasibility of A in $\left(\prod_{i=1}^k a_i\right)^{O(1)} = k^{O(2^k)}$ time by Claim 18. Otherwise, let $\ell \in [k-1]$ be the minimum integer with $a_{\ell+1} > k^{2^{\ell}}$, which can be computed in $O(n)$ time. Then, we obtain

$$a_{\ell+1} > k^{2^{\ell}} = k \prod_{i=1}^{\ell} k^{2^{i-1}} > (k-\ell) \prod_{i=1}^{\ell} a_i \quad (10)$$

in the same way as (5). We observe that $(a_i)_{i \in [k]}$ is a yes-instance if and only if $(a_i)_{i \in [\ell]}$ is a strong yes-instance, as the “only if” part is trivial and the “if” part follows from Claim 20 and (10). Therefore, it suffices to determine whether $(a_i)_{i \in [\ell]}$ is a strong yes-instance or not, which can be done in $\left(\prod_{i=1}^{\ell} a_i\right)^{O(1)}$ time by Claim 18. This running time is bounded by $k^{O(2^k)}$, because $a_i \leq k^{2^{i-1}}$ for $i \in [\ell]$. $\blacktriangleleft$