Electronic Colloquium on Computational Complexity, Report No. 126 (2024)

On the Complexity of Some Restricted Variants of QUOTIENT PIGEON and a Weak Variant of KÖNIG

Takashi Ishizuka
Artificial Intelligence Laboratory, Fujitsu Limited, Japan
ishizuka-t@fujitsu.com

June 17, 2024

Abstract

One of the most famous TFNP subclasses is PPP, which is the set of all search problems whose totality is guaranteed by the pigeonhole principle. The author's recent preprint [Ish24] has introduced a TFNP problem related to the pigeonhole principle over a quotient set, called QUOTIENT PIGEON, and shown that the problem QUOTIENT PIGEON is not only PPP-hard but also PLS-hard. In this paper, we formulate other computational problems related to the pigeonhole principle over a quotient set via an explicit representation of the equivalence classes. Our new formulation introduces a non-trivial $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete problem for some $k \geq 2$. Furthermore, we consider the computational complexity of a computational problem related to König's lemma, which is a weaker variant of the problem formulated by Pasarkar, Papadimitriou, and Yannakakis [PPY23]. We show that our weaker variant is PPAD-hard and is in $\mathtt{PPP} \cap \mathtt{PPA}$.

Keywords: TFNP, PPP, the pigeonhole principle, König's lemma

ISSN 1433-8092

1 Introduction

Megiddo and Papadimitriou [MP91] have begun to study the computational complexity of search problems that always have at least one solution. We call such problems *total* search problems. The complexity class `TFNP` [MP91; Pap94] captures the computational aspects of search problems whose existence of solutions is guaranteed, and its correctness is effortlessly checkable. We know that many significantly important computational problems belong to `TFNP`; for example, finding a Nash equilibrium [CDT09; DGP09], computing a fair division [FG18; DFM22; GHH23], integer factoring [Bur06; Jer16], and algebraic problems related to cryptographies [SZZ18; HV21].

A natural way to analyze the theoretical features of a complexity class is to characterize its class by complete problems. However, it is widely believed that `TFNP` has no complete problem [Pud15; Pap94]. Consequently, several `TFNP` subclasses with complete problems have been introduced over the past three decades. The following four classes are the best well-known `TFNP` subclasses.

`PLS` [JPY88] Every finite directed acyclic graph has a sink.

`PPA` [Pap94] Every finite undirected graph with a known odd-degree node must have another odd-degree node. In other words, the existence of another solution is guaranteed by the handshaking lemma.

`PPAD` [Pap94] Every finite directed graph with a known unbalanced node must have another unbalanced node, where "*unbalanced*" means that out-degree $\neq$ in-degree.

`PPP` [Pap94] Every function that maps N elements to $N-1$ elements must have a collision. In other words, the existence of a solution is guaranteed by the pigeonhole principle.

In this paper, we shed light on the computational complexity of the pigeonhole principle over a quotient set, which is a generalization of the canonical `PPP`-complete problem Pigeon (see Definition 1). As mentioned above, the complexity class `PPP` is one of Papadimitriou's traditional `TFNP` subclasses [Pap94]. Roughly, this class is formulated as the class for search problems related to the *pigeonhole principle*, i.e., the class for problems whose totality is guaranteed by the pigeonhole principle.

The formal definition of the canonical `PPP`-complete problem is as follows.

Definition 1. Pigeon
Input:

- A Boolean circuit $C : \{0,1\}^n \to \{0,1\}^n$
- A special element $v^* \in \{0,1\}^n$

Output: one of the following:

(1) two distinct elements $x, y \in \{0,1\}^n$ such that $C(x) = C(y)$

(2) an element $x \in \{0,1\}^n$ such that $C(x) = v^*$

Definition 2 (Class `PPP`)**.** The complexity class `PPP` is the set of all search problems that are reducible to Pigeon in polynomial time.

There are some studies to consider the computational complexity of a generalization of the pigeonhole principle. Pasarkar, Papadimitriou, and Yannakakis [PPY23] have introduced the complexity class `PLC` (and also `UPLC`), which captures the computational aspects of the *iterative* use of the pigeonhole principle. They have proven that the class `PLC` contains a computational problem related to Ramsey's theorem. Jain, Li, Robere, and Xun [Jai+24] have investigated the generalization of the class `PPP` and its hierarchy called "*Pecking Order*." The pecking order principle, a *generalized pigeonhole principle*, states that for $t \geq 2$, if $(t-1)N+1$ pigeons map to N holes, then there is a hole that contains t pigeons. They have introduced the new `TFNP` subclasses the Pigeon Hierarchy (`PiH`), `SAP`, and `PAP`, and showed that the class `PAP` contains `UPLC`. Another recent related work by Fleming, Grosser, Pitassi, and Robere [Fle+24] has shown that, in the black-box setting, the class `PPP` is not closed under the Turing reduction.

The author's recent preprint [Ish24] has introduced another generalization of PIGEON, called QUOTIENT PIGEON, which is `PPP`- and `PLS`-hard. In the approach shown in [Ish24], we consider a PIGEON instance over the quotient set $U/\sim$, where U is a finite set and $\sim$ is an equivalence relation over U. Thus, we focus on the following search problem: Given a function $C : U/\sim \to U/\sim$ and a special element v^* in U, find two distinct elements $x, y \in U/\sim$ such that $C(x) \sim C(y)$ or an element $x \in U/\sim$ such that $C(x) \sim v^*$.

To formulate the above variant of PIGEON, we allow to obtain another function $E : U \times U \to \{0,1\}$ computing an equivalence relation over U. We denote by $\sim_E$ the binary relation defined by E; for each pair of elements x, y in U, $x \sim_E y$ if and only if $E(x,y) = 1$. Formally, the new search problem called QUOTIENT PIGEON is defined as follows.

Definition 3. QUOTIENT PIGEON
Input:

- Two Boolean circuits $C : \{0,1\}^n \to \{0,1\}^n$ and $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$
- An element $v^* \in \{0,1\}^n$

Output: one of the following:

(1) two elements $x, y \in \{0,1\}^n$ such that $x \not\sim_E y$ and $C(x) \sim_E C(y)$

(2) an element $x \in \{0,1\}^n$ such that $C(x) \sim_E v^*$

(3) two elements $x, y \in \{0,1\}^n$ such that $x \sim_E y$ and $C(x) \not\sim_E C(y)$

(4) an element $x \in \{0,1\}^n$ such that $E(x,x) = 0$

(5) two elements $x, y \in \{0,1\}^n$ such that $E(x,y) \neq E(y,x)$.

(6) three distinct elements $x, y, z \in \{0,1\}^n$ such that $x \sim_E y$, $y \sim_E z$, and $x \not\sim_E z$

Unfortunately, we are unaware of a way of syntactically enforcing the Boolean circuit E to compute an equivalence relation over the finite set $\{0,1\}^n$. Thus, we introduce violations as solutions to QUOTIENT PIGEON to ensure that this problem belongs to `TFNP`. More precisely, the fourth-type solution is a violation of the *reflexivity*. The fifth-type solution represents a violation of the *symmetry*. Finally, the sixth-type solution means a violation of the *transitivity*.

We now introduce a new `TFNP` subclass, $\mathtt{PPP}/\sim$, which is a set of all search problems that are reducible to QUOTIENT WEAK PIGEON in polynomial time. Ishizuka [Ish24] has proven that QUOTIENT PIGEON is not only $\mathtt{PPP}$-hard but also $\mathtt{PLS}$-hard.

Definition 4 (Class $\mathtt{PPP}/\!\sim$)**.** The complexity class $\mathtt{PPP}/\!\sim$ is the set of all search problems that are reducible to QUOTIENT PIGEON in polynomial time.

Theorem 5 ([Ish24])**.** $\mathtt{PLS} \cup \mathtt{PPP} \subseteq \mathtt{PPP}/\!\sim\, \subseteq \mathtt{TFNP}$.

1.1 Our Contributions and Paper Organization

In this section, we summarize our contribution and sketch some proofs. We will present some notations that are used in this paper in Section 2. In this paper, we discuss the computational complexity of some variants of PIGEON over quotient sets. Recall that the $\mathtt{PLS}$-hardness proof of QUOTIENT PIGEON shown in [Ish24] heavily relies on the implicit feature of equivalence classes. In a QUOTIENT PIGEON instance, we can efficiently verify whether given two elements belong to the same equivalence class. However, it does not guarantee to effortlessly find another element belonging to the same class.

In this paper, we investigate some variants of QUOTIENT PIGEON whose equivalence classes have an explicit feature, i.e., we can efficiently find elements in the same class with a given element. We show that such a variant induces a $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete problem for some positive integer $k \geq 2$.

Explicit & Well-Balanced Variants We consider easier variants of QUOTIENT PIGEON in which, for a given element x, we can efficiently get another element that is equal to x. To formulate such a variant, we obtain a Boolean circuit $L : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$ that returns all elements in the same equivalence class with the input element instead of the Boolean circuit E that decide whether the given two elements are equivalent. For a list function $L : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$, we denote by $\sim_L$ the binary relation defined by L: For each pair of elements x, y in $\{0,1\}^n$, $x \sim_L y$ if and only if $x \in L(y)$ and $y \in L(x)$.

Definition 6. EXPLICIT QUOTIENT PIGEON
Input:

- Two Boolean circuits $C : \{0,1\}^n \to \{0,1\}^n$ and $L : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$
- An element $v^* \in [2^n]$

Output: one of the following:

(1) two elements $x, y \in \{0,1\}^n$ such that $x \not\sim_L y$ and $C(x) \sim_L C(y)$

(2) an element $x \in \{0,1\}^n$ such that $C(x) \sim_L v^*$

(3) two elements $x, y \in \{0,1\}^n$ such that $x \sim_L y$ and $C(x) \not\sim_L C(y)$

(4) an element $x \in \{0,1\}^n$ such that $x \in L(x)$

(5) two elements $x, y \in \{0,1\}^n$ such that $x \in L(y)$ and $y \notin L(x)$.

Similarly to QUOTIENT PIGEON, we are unaware of a way of syntactically enforcing the Boolean circuit E to compute an equivalence relation over the finite set $\{0,1\}^n$. Thus, we introduce violations as solutions to EXPLICIT QUOTIENT PIGEON to ensure that this problem belongs to TFNP. More precisely, the fourth-type solution is a violation of the reflexivity. The fifth-type solution represents a violation of the symmetry. Note that the transitivity is guaranteed by the combination of fourth- and fifth-type solutions to this problem.

We prove that Explicit Quotient Pigeon is PPP-complete. The PPP-hardness of this problem is oblivious. Hence, It suffices to show that we have a polynomial-time reduction from Explicit Quotient Pigeon to the canonical PPP-complete problem, Pigeon.

Theorem 7. Explicit Quotient Pigeon *is* PPP*-complete.*

Proof. Let two Boolean circuits $C : \{0,1\}^n \to \{0,1\}^n$ and $L : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$ and an element v^* be a Explicit Quotient Pigeon instance, where k is some positive integer. We now construct a Pigeon instance $f : \{0,1\}^n \to \{0,1\}^n$. Our idea is simple. Let x be any element in $\{0,1\}^n$. We denote by $\xi_1, \ldots, \xi_d$ the elements equivalences to x, i.e., $\xi_j \sim_L x$ for each $i \in [d]$. Suppose the elements $\xi_1, \ldots, \xi_d$ are sorted in lexicographic order: $\xi_1 \prec \cdots \prec \xi_d$. Then, define $f(\xi_i) := \xi_{i+1}$ for each $i \in [d-1]$. Finally, we define $f(\xi_d) := \eta_1$, where η_1 is the element that is the lexicographically least one among the elements in the same equivalence class as $C(\xi_d)$. It is easy to see that we can obtain a solution to the original Explicit Quotient Pigeon instance from each solution to the reduced Pigeon instance in polynomial time. □

From the above observation, the explicit variant of Quotient Pigeon does not change the computational intractability of the pigeonhole principle. Therefore, we focus on the complexity of the variant of Quotient Pigeon that has a further restriction. Recall that, in the reduction from Pigeon to Quotient Pigeon shown in [Ish24], we use a simple equivalence relation: All elements are different from each other. In other words, the size of every equivalence class is exactly one. Now, we introduce a further restricted variant of Explicit Quotient Pigeon in which every equivalence class has the same size. Naturally, a variant that we simply added a size violation as a solution is also PPP-complete. To formulate a more easier problem than Pigeon, we ensure the existence of size violations.

Our new variant of Explicit Quotient Pigeon also belongs to $\mathtt{PPA}_k$ for some positive integer k. When $k = 2$, the complexity class is the same as one of the traditional TFNP subclasses, PPA, which is the set of all search problems whose totality is guaranteed by the handshaking lemma. Consider an instance of Explicit Quotient Pigeon $\langle C : \{0,1\}^n \to \{0,1\}^n; L : \{0,1\}^n \to \mathrm{Set}_k(\{0,1\}^n); v^*\rangle$. We write $m(v^*)$ for the number of elements such that $x \sim_L v^*$. We know that there is an unknown equivalence class whose size is strictly less than k if $2^n - m(v^*) \neq 0 \mod k$. Hence, we consider the problem of finding a solution to Explicit Quotient Pigeon or a violation of the size condition. We call such a variant k-Well-Balanced Explicit Quotient Pigeon, and we prove this total search problem is $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete.

Definition 8. k-Well-Balanced Explicit Quotient Pigeon
Input:

- Two Boolean circuits $C : \{0,1\}^n \to \{0,1\}^n$ and $L : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$
- An element $v^* \in \{0,1\}^n$

Output: one of the following:

(1) two elements $x, y \in \{0,1\}^n$ such that $x \not\sim_L y$ and $C(x) \sim_L C(y)$

(2) an element $x \in \{0,1\}^n$ such that $C(x) \sim_L v^*$

(3) two elements $x, y \in \{0,1\}^n$ such that $x \sim_L y$ and $C(x) \not\sim_L C(y)$

(4) an element $x \in \{0,1\}^n$ such that $x \in L(x)$

(5) two elements $x, y \in \{0,1\}^n$ such that $x \in L(y)$ and $y \notin L(x)$.

(6) an element $x \in \{0,1\}^n$ such that $x \sim_L v^*$ and $2^n - |L(x)| = ck$ for some integer c.

(7) an element $x \in \{0,1\}^n$ such that $x \not\sim_L v^*$ and $|L(x)| < k$

Theorem 9. *Let $k \geq 2$ be a positive integer. The problem k-Well-Balanced Explicit Quotient Pigeon is $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete.*

Weak Variant We also consider the quotient variant of Weak Pigeon, which is a canonical PWPP-complete problem. The total search problem Weak Pigeon is defined as follows: Given a Boolean circuit $C : \{0,1\}^n \to \{0,1\}^m$, where $m < n$, find two distinct elements $x, y \in \{0,1\}^n$ such that $C(x) = C(y)$.

Definition 10. Quotient Weak Pigeon
Input: Two Boolean circuits $C : \{0,1\}^n \to \{0,1\}^{n-1}$ and $M : \{0,1\}^n \to \{0,1\}^n$ and a special element $v^* \in \{0,1\}^n$ such that $M(v^*) = v^*$
Output: one of the following:

(1) two elements $x, y \in \{0,1\}^n$ such that $x \not\sim_M y$, and $C(x) = C(y)$; and

(2) an element $x \in \{0,1\}^n$ such that $M(x) \neq x$ and $M(M(x)) \neq x$,

where we define $x \sim_M y$ if and only if $x = y$ or $M(x) = y$ and $M(y) = x$.

Interestingly, the problem Quotient Weak Pigeon is PPP-complete. The PPP-containment of this total search problem follows from the Pecking Order Principle. Assuming that the matching function M is an involution, there are at least $2^{n-1} + 2$ pigeons. Then, we map these pigeons to 2^{n-1} holes according to the hole function C. This implies that there are two pigeons that are mapped to the same hole. Hence, the challenging part is to prove the PPP-hardness of Weak Quotient Pigeon. To prove this fact, we introduce dummy nodes and define a good matching function. The full proof of Theorem 11 can be found in Section 3.2

Theorem 11. Quotient Weak Pigeon *is* PPP*-complete.*

Next, we consider the computational complexity of the well-balanced variant of Weak Quotient Pigeon. Recall that in the definition of Weak Quotient Pigeon we require that the special element v^* is an isolated node. This implies that there is another isolated node from the handshaking lemma. Our well-balanced variant allows another isolated node as a solution. We call this variant Well-Balanced Quotient Weak Pigeon; the formal definition can be found in Definition 12.

Definition 12. Well-Balanced Quotient Weak Pigeon
Input:

- Two Boolean circuits $C : \{0,1\}^n \to \{0,1\}^{n-1}$ and $M : \{0,1\}^n \to \{0,1\}^n$
- A special element $v^* \in \{0,1\}^n$ such that $M(v^*) = v^*$

Output: one of the following:

(1) two elements $x, y \in \{0,1\}^n$ such that $x \not\sim_M y$, and $C(x) = C(y)$;

(2) an element $x \in \{0,1\}^n \setminus \{v^*\}$ such that $M(x) = x$; and

(3) an element $x \in \{0,1\}^n$ such that $M(x) \neq x$ and $M(M(x)) \neq x$,

where we define $x \sim_M y$ if and only if $x = y$ or $M(x) = y$ and $M(y) = x$.

By definition, the problem Well-Balanced Quotient Weak Pigeon also belongs to the complexity class PPA. In Section 3.3, we prove that Well-Balanced Quotient Weak Pigeon is PPAD-hard.

Theorem 13. *The following two statements hold:*

(i) Well-Balanced Quotient Weak Pigeon *belongs to* $\mathtt{PPP} \cap \mathtt{PPA}$*.*

(ii) The problem Well-Balanced Quotient Weak Pigeon *is* PPAD*-hard.*

Applications Finally, we present an application of the problem Pigeon over quotient sets.

Pasarkar, Papadimitriou, and Yannakakis [PPY23] have formulated a computational problem related to König's lemma, which states that every infinite rooted tree with finite branching has an infinite path starting at the root. They have focused on the finite version of this lemma: Every rooted binary tree with 2^n nodes contains a path of length n starting at the root. In their problem, called König, we are given a rooted binary tree with 2^n nodes via two Boolean circuits: One returns a parent of a node, and the other indicates a node type.

Consider two Boolean circuits $P : \{0,1\}^n \to \{0,1\}^n$ and $C : \{0,1\}^n \to \{0,1\}$; we suppose that these circuits define a rooted binary tree on 2^n nodes, where each node is encoded by an n-bit binary string. For each node u, $P(u)$ is a parent of u, and the circuit C indicates whether the node u is a left or right child of $P(u)$. Here, we interpret the node u as a left child of $P(u)$ if $C(u) = 0$; otherwise, we interpret u as a right child of $P(u)$. In the problem König, we are also given a root r that satisfies $P(r) = r$. Then, the task of this search problem is to find either a path of length n or a violation that two circuits P and C do not specify a connected binary tree rooted at r.

Pasarkar, Papadimitriou, and Yannakakis [PPY23] have proven the PPP-completeness of König. The original formulation has an implicit feature; that is, it may be hard to compute a sibling for a given node. This paper considers the computational complexity of an explicit variant of König, called Weak König. In our problem, we are given two Boolean circuits computing a parent and a sibling, respectively. Then, the goal of the problem is the same, i.e., to find either a path of length n or a violation.

The formal definition of Weak König can be found in Definition 14. Here, the first type of solution to Weak König is called a *Long Path* certificate. The second-type, third-type, and fifth-type solutions represent a violation of the parent-sibling condition. The fourth type of solution is a *Non-Unique Root* witness (i.e., $P(x) = x$) or a node without siblings. Finally, the sixth type of solution is called a *Far Away* certificate.

In Section 3.4, we show that Weak König and Quotient Weak Pigeon are polynomial-time reducible to each other. Hence, the total search problem Weak Pigeon is PPAD-hard and belongs to PPP ∩ PPA from Theorem 13.

Definition 14. Weak König
Input:

- Two Boolean circuits $P : \{0,1\}^n \to \{0,1\}^n$ and $S : \{0,1\}^n \to \{0,1\}^n$
- A root node $r \in \{0,1\}^n$ such that $P(r) = r = S(r)$

Output: one of the following:

(1) an element $x \in \{0,1\}^n$ that yields the following sequence $a_0, \ldots, a_n$ such that $a_0 := x$ and $a_i := P(a_{i-1})$ for each $i \in [n]$, $a_i \neq a_j$ for all $0 \leq i < j \leq n$, and $a_n = r$;

(2) two distinct elements $x, y \in \{0,1\}^n$ such that $P(x) = P(y)$ and $S(x) \neq S(y)$

(3) two distinct elements $x, y \in \{0,1\}^n$ such that $P(x) \neq P(y)$ and $S(x) = S(y)$;

(4) an element $x \in \{0,1\}^n \setminus \{r\}$ such that $P(x) = x$ or $S(x) = x$;

(5) an element $x \in \{0,1\}^n$ such that $S(x) \neq x$ and $S(S(x)) \neq x$;

(6) an element $x \in \{0,1\}^n$ such that $P^{n+1}(x) \neq r$.

Theorem 15. Weak König *is polynomially equivalent to* Quotient Weak Pigeon.

1.2 Conclusion and Open Questions

This paper has investigated the computational complexity of the pigeonhole principle over quotient sets when the equivalence class is explicitly represented. We have introduced a $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete problem k-Well-Balanced Explicit Quotient Pigeon. To our best knowledge, this is the first non-trivial $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete problem; we answer an open question pointed out in [HKT24]. We have also studied the complexity of weak variants of Quotient Pigeon and shown that Well-Balanced Quotient Weak Pigeon lies between PPAD and PPP ∩ PPA.

This paper has left the following open questions:

> Is Well-Balanced Quotient Weak Pigeon PPAD-complete or PPP ∩ PPA-complete?

Note that proving the $\mathtt{PPP} \cap \mathtt{PPA}$-completeness of Well-Balanced Quotient Weak Pigeon leads us to a natural $\mathtt{PPP} \cap \mathtt{PPA}$-complete problem Weak König from Theorem 15. On the other hand, showing the $\mathtt{PPAD}$-completeness of this problem improves the upper bound of integer factoring under randomized reductions among Papadimitriou's traditional $\mathtt{TFNP}$ subclasses. Recall that the complexity class $\mathtt{PWPP}$ is the set of all search problems that are polynomial-time reducible to Weak Pigeon. By the definition of Well-Balanced Quotient Weak Pigeon, we can straightforwardly see that this search problem is $\mathtt{PWPP} \cap \mathtt{PPA}$-hard. Finally, we know that there is a polynomial-time randomized reduction from integer factoring to a problem in $\mathtt{PWPP} \cap \mathtt{PPA}$ [Bur06; Jer16]. Hence, showing the $\mathtt{PPAD}$-completeness of Well-Balanced Quotient Weak Pigeon implies that there is a polynomial-time randomized reduction from integer factoring to a problem in $\mathtt{PPAD}$.

Another research direction worth considering is to study the computational complexity of quotient variants of End of Line and End of Potential Line, which is a $\mathtt{PPAD} \cap \mathtt{PLS}$-complete problem [Göö+22]. By using the same approach shown in this paper or [Ish24], we can easily introduce quotient variants of these problems. We have an interesting question related to this research direction: Do quotient variants of End of Line and End of Potential Line help us to capture the complexity of the variants with super-polynomially many known sources of End of Line and End of Potential Line, respectively? It is still open whether such variants are also $\mathtt{PPAD}$- and $\mathtt{EOPL}$-complete, respectively [HG18; Ish21].

2 Preliminaries

We denote by $\mathbb{Z}$ the set of all integers. For an integer $a \in \mathbb{Z}$, we define $\mathbb{Z}_{\geq a} := \{x \in \mathbb{Z} \ : \ x \geq a\}$ and $\mathbb{Z}_{>a} := \{x \in \mathbb{Z} \ : \ x > a\}$. We use $[n] := \{1, 2, \dots, n\}$ and $[m, n] := \{m, m+1, \dots, n-1, n\}$ for every positive integer n in $\mathbb{Z}_{>0}$ and every non-negative integer m with $m \leq n$. Let X be a finite set. We denote by $|X|$ the cardinality of the elements in X.

Let $\{0,1\}^*$ denote the set of binary strings with a finite length. For every string $x \in \{0,1\}^*$, we denote by $|x|$ the length of x. For each positive integer n, we write $\{0,1\}^n$ for the set of binary strength with the length n. Throughout this paper, we sometimes regard $\{0,1\}^n$ as the set of non-negative integers in $[0, 2^n - 1]$.

Many of the computational problems appearing in this paper involve Boolean circuits whose output is interpreted as a set. We will use the same way of encoding set as used in [Hol21]. We denote by $C : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$ the Boolean circuit that takes as input an n-bit string and outputs a set of at most k strings in $\{0,1\}^n$. More precisely, the Boolean circuit $C : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$ has n input bits and $kn+1$ output bits. Here, we use the following encoding: Each set $\{\xi_1, \dots, \xi_\ell\} \subseteq \{0,1\}^n$, where $\ell \leq k$, is represented by the string $\xi_1 \sharp \cdots \sharp \xi_\ell \sharp 1 \sharp 0^{(k-\ell)n} \in \{0,1\}^{kn+1}$. We can effortlessly verify whether a string in $\{0,1\}^{kn+1}$ is a valid representation of a set and if not, we just interpret it as an empty set.

Search Problems Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be a relation. We say that R is *polynomially balanced* if there is a polynomial $p : \mathbb{Z}_{\geq 0} \to \mathbb{Z}_{\geq 0}$ such that for each $(x,y) \in R$, it holds that $|y| \leq p(|x|)$. We say that R is *polynomial-time decidable* if for each pair of strings $(x,y) \in \{0,1\}^* \times \{0,1\}^*$, we can decide whether (x,y) belongs to R in polynomial time. We say that R is *total* if for every string $x \in \{0,1\}^*$, there always exists at least one string y such that $(x,y) \in R$.

For a relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$, the search problem with respect to R is defined as follows[1]: Given a string $x \in \{0,1\}^*$, find a string $y \in \{0,1\}^*$ such that $(x, y) \in R$ if such a y exists, otherwise reports "*no*." When R is also total, we call such a search problem a total search problem. The complexity class FNP is the set of all search problems with respect to a polynomially balanced and polynomial-time decidable relation R. The complexity class TFNP is the set of all total search problems belonging to FNP. By definition, it holds that TFNP $\subseteq$ FNP.

Reductions Let $R, S \subseteq \{0,1\}^* \times \{0,1\}^*$ be two search problems. A polynomial-time reduction from R to S is defined by two polynomial-time computable functions $f : \{0,1\}^* \to \{0,1\}^*$ and $g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ satisfying that $(x, g(x, y)) \in R$ whenever $(f(x), y) \in S$. In other words, the function f maps an instance x of R to an instance $f(x)$ of S, and the other function g maps a solution y to the instance $f(x)$ to a solution $g(x, y)$ to the instance x.

2.1 Complexity Classes

Class $\mathtt{PPA}_k$ We now recall the definition of the class $\mathtt{PPA}_k$ for some positive integer $k \geq 2$. Roughly, this class is a modulo-k analog of the well-known TFNP subclass PPA. There are many ways to define the class $\mathtt{PPA}_k$. In this paper, we use the search problem $\textsc{Lonely}_k$ (see Definition 16) to formulate this class. The complexity class $\mathtt{PPA}_k$ is a set of all search problems that are reducible to $\textsc{Lonely}_k$ in polynomial time [Hol21].

Definition 16. $\textsc{Lonely}_k$
Input:

- a Boolean circuit $L : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n)$
- a set $\Pi^* \subseteq \{0,1\}^n$ such that $2^n - |\Pi^*| \neq 0 \mod k$

Output: one of the following:

(1) an element $x \in \{0,1\}^n \setminus \Pi^*$ such that $|L(x)| < k$;

(2) an element $x \in \{0,1\}^n$ such that $x \notin L(x)$; and

(3) two elements $x, y \in \{0,1\}^n$ such that $y \in L(x)$ and $x \notin L(y)$.

Class PPAD Finally, we recall the complexity class PPAD, introduced by Papadimitriou [Pap94]. Of course, we have many ways to define the class PPAD. Here, we use the search problem End of Line (see Definition 17), which is the canonical PPAD-complete problem. Hence, the class PPAD is the set of all search problems that are reducible to End of Line in polynomial time.

[1]For simplicity, we call the search problem with respect to R the search problem R.

Definition 17. End of Line
Input:

- a Boolean circuit $S, P : \{0,1\}^n \to \{0,1\}^n$ with $S(0^n) \neq 0^n = P(0^n)$

Output: one of the following:

(1) a sink $x \in \{0,1\}^n$ such that $P(S(x)) \neq x$; and

(2) an unkonwn source $x \in \{0,1\}^n$ such that $S(P(x)) \neq x \neq 0^n$.

3 Proofs of Our Results

3.1 $\mathtt{PPP} \cap \mathtt{PPA}_k$-completeness of k-Well-Balanced Explicit Quotient Pigeon

This section proves Theorem 9, that is, the $\mathtt{PPP} \cap \mathtt{PPA}_k$-completeness of k-Well-Balanced Explicit Quotient Pigeon. By definition, the $\mathtt{PPP} \cap \mathtt{PPA}_k$ membership of this total search problem is trivial. Hence, we show the $\mathtt{PPP} \cap \mathtt{PPA}_k$-hardness of k-Well-Balanced Explicit Quotient Pigeon. To prove the $\mathtt{PPP} \cap \mathtt{PPA}_k$-hardness, we focus on the canonical $\mathtt{PPP} \cap \mathtt{PPA}_k$-complete problem, EitherSolution(Pigeon, Lonely$_k$): Given a pair of instances of Pigeon and Lonely$_k$, find a solution to *either* Pigeon or Lonely$_k$.

Let $\langle f : \{0,1\}^n \to \{0,1\}^n; v^* \rangle$ and $\langle g : \{0,1\}^n \to \mathrm{Set}_{\leq k}(\{0,1\}^n); \Pi^* \rangle$ be instances of Pigeon and Lonely$_k$, respectively. We will construct an instance of k-Well-Balanced Explicit Quotient Pigeon. Our idea is simple. We create k copies of the Pigeon instance. Each element on the Pigeon instance forms an equivalence class with its copies; this technique naturally induces equivalence classes that are the same size. On the other hand, each element on the instance of Lonely$_k$, except for every element in Π^*, is a fixed point in the reduced instance. We map each element in Π^* to the special element v^* of the Pigeon instance. Finally, we specify the special element of the reduced k-Well-Balanced Explicit Quotient Pigeon instance with an element in Π^*.

3.2 $\mathtt{PPP}$-completeness of Quotient Weak Pigeon

In this section, we show that the $\mathtt{PPP}$-completeness of Quotient Weak Pigeon. First, we prove the $\mathtt{PPP}$-hardness of Quotient Weak Pigeon. Consider the Pigeon instance $\langle C : \{0,1\}^n \to \{0,1\}^n; v^* \rangle$. Without loss of generality, we can assume that there is no element $x \in \{0,1\}^n$ such that $C(x) \neq v^*$. We now construct a Quotient Weak Pigeon instance $\langle f : \{0,1\}^{1+n} \to \{0,1\}^n; M : \{0,1\}^{1+n} \to \{0,1\}^{1+n}; u^* \rangle$.

We define the matching function M as follows: For each $b \in \{0,1\}$ and each $x \in \{0,1\}^n$,

$$M(b \sharp x) := \begin{cases} b \sharp x & \text{if } x = v^*, \\ (1-b) \sharp x & \text{if } x \neq v^*. \end{cases} \tag{1}$$

We set the special element to be $u^* = 0 \sharp v^* \in \{0,1\}^{1+n}$. Finally, we define the Boolean circuit f as follows: For each $b \in \{0,1\}$ and each $x \in \{0,1\}^n$,

$$f(b\sharp x) := \begin{cases} v^* & \text{if } b = 1 \text{ and } x = v^*, \\ C(x) & \text{otherwise.} \end{cases} \tag{2}$$

We complete the construction of the reduced instance. What remains is to show that we can efficiently obtain an original solution from every solution to the reduced instance.

By the definition of the matching function M, there is no second-type solution to Quotient Weak Pigeon. Hence, we only obtain the first-type solution, i.e., two elements $b_x\sharp x, b_y\sharp y \in \{0,1\}^{1+n}$ such that $b_x\sharp x \not\sim_M b_y\sharp y$ and $f(b_x\sharp x) = f(b_y\sharp y)$. From our assumption, we know that $f(b_x\sharp x) \neq v^*$, and thus, we have that $b_x\sharp x \neq 1\sharp v^*$ and $b_y\sharp y \neq 1\sharp v^*$. This fact also implies that $x \neq y$. By definition of the function f, it holds that $C(x) = C(y)$. Hence, the pair of two elements (x, y) is a solution to the original Pigeon instance.

Next, we show that there is a polynomial-time reduction from Quotient Weak Pigeon to Pigeon. Let $\langle C : \{0,1\}^n \to \{0,1\}^{n-1}; M : \{0,1\}^n \to \{0,1\}^n; v^* \rangle$ be any instance of Quotient Weak Pigeon. It is not hard to see that we can assume that there is no second-type solution to Quotient Weak Pigeon without loss of generality. We redefine the function M as follows: For every $x \in \{0,1\}^n$,

$$M'(x) := \begin{cases} x & \text{if } M(x) \neq x \text{ and } M(M(x)) \neq x, \\ M(x) & \text{otherwise.} \end{cases} \tag{3}$$

Before starting the reduction to Pigeon, we observe that, without loss of generality, we can assume the following two properties:

1. $v^* = 0^n$ and
2. $C(v^*) = 0^{n-1}$.

The first property follows the following reconstruction of the matching function M. For every $x \in \{0,1\}^n$,

$$M'(x) := \begin{cases} 0^n & \text{if } x = 0^n, \\ M(0^n) & \text{if } x = v^*, \\ v^* & \text{if } M(x) = 0^n, \\ M(x) & \text{otherwise.} \end{cases} \tag{4}$$

The second property follows the following reconstruction of the function C. For every $x \in \{0,1\}^n$,

$$C'(x) = \begin{cases} 0^{n-1} & \text{if } x = v^* \text{ or } f(x) = f(v^*), \\ C(v^*) & \text{if } x \neq v^* \text{ and } f(x) = 0^{n-1}, \\ C(x) & \text{otherwise.} \end{cases} \tag{5}$$

We will construct a Pigeon instance $\langle f : \{0,1\}^n \to \{0,1\}^n, v^* \rangle$. The function f is defined as follows: For each $x \in \{0,1\}^n$,

$$f(x) := \begin{cases} 1\sharp C(v^*) & \text{if } x = v^*, \\ 1\sharp C(x) & \text{if } M(x) = x, \\ 1\sharp C(x) & \text{if } M(x) \neq x \text{ and } x \prec M(x), \\ 0\sharp C(x) & \text{othwerwise}, \end{cases} \tag{6}$$

where $x \prec M(x)$ means that x is less than $M(x)$ under the lexicographic order.

We complete the construction of the Pigeon instance. What remains is to show that we can efficiently obtain a solution to the original instance from every solution to the reduced instance.

First, we consider the case where we obtain an element $x \in \{0,1\}^n$ such that $f(x) = v^* = 0^n$. In this case, we know there is another element y such that $M(x) = y$. By definition, it holds that $C(y) = 0^{n-1}$ and $y \not\sim_M v^*$. From our assumption, $C(v^*) = 0^{n-1}$. Therefore, a pair of two elements $(M(x), v^*)$ is a solution to the original Quotient Weak Pigeon instance.

Next, we consider the case where we obtain two elements $x, y \in \{0,1\}^n$ such that $x \neq y$ and $f(x) = f(y)$. We first observe $x \not\sim_M y$. If $x \sim_M y$, then the first bits of $f(x)$ and $f(y)$ are different; this is a contradiction that a pair of elements (x, y) is a collision. By the definition of f, it satisfies that $C(x) = C(y)$. Hence, a pair of elements (x, y) is a solution to the original instance of Quotient Weak Pigeon.

3.3 On the Complexity of Well-Balanced Quotient Weak Pigeon

In this section, we prove Theorem 13. It is not hard to see that the problem Well-Balanced Quotient Weak Pigeon belongs to $\mathtt{PPP} \cap \mathtt{PPA}$. Therefore, the main part of this section is to show the $\mathtt{PPAD}$-hardness of Well-Balanced Quotient Weak Pigeon. In other words, we prove the second part of Theorem 13.

We will construct a polynomial-time reduction from End of Line to Well-Balanced Quotient Weak Pigeon. Let two Boolean circuits $S, P : \{0,1\}^n \to \{0,1\}^n$ with $S(0^n) \neq 0^n = P(0^n)$ be any instance of End of Line. We now construct two Boolean circuits $C : \{0,1\} \times \{0,1\}^n \to \{0,1\} \times \{0,1\}^n$ and $M : \{0,1\} \times \{0,1\}^n \to \{0,1\} \times \{0,1\}^n$, which are a Quotient Weak Pigeon instance.

Before starting our reduction, we outline our proof idea. We denote by $G = (V, E)$ the digraph defined by two Boolean circuits S, P. We create a dummy node v' for each node v in V. For each valid arc (v, u) in E, we replace this with two arcs (v, v') and (v', u). For every sink node $v \in V$, we create a new arc (v, v'). We obtain a new digraph G' with $2|V|$ nodes such that for each sink node on G, its dummy is a sink node on G'. Then, we define the matching M as follows: For an original node v, $M(v)$ returns its predecessor, and for a dummy node v', $M(v')$ returns its successor. Thus, every source node and every sink node on G' are isolated points. Finally, we define the hole function C as follows: For each original node v, $C(v)$ returns itself, and for each dummy node v', $C(v')$ returns its matching node if such a node exists; otherwise $C(v')$ returns the distinguished source node 0^n. From the construction, it is not hard to see that we can obtain a solution to End of Line from every solution to Quotient Weak Pigeon.

Let us move on to the formal definition of our reduction. Without loss of generality, we can assume that two Boolean circuits S and P compute valid arcs. For each pair of $b \in \{0,1\}$ and $x \in \{0,1\}^n$, we define the Boolean circuits M and C as follows:

$$M(b,x) := \begin{cases} (b,x) & \text{if either } b = 1 \text{ and } x \text{ is a sink or } b = 0 \text{ and } x \text{ is a source;} \\ (0, S(x)) & \text{if } b = 1 \text{ and } x \text{ is not a sink;} \\ (1, P(x)) & \text{if } b = 0 \text{ and } x \text{ is not a source.} \end{cases}$$

$$C(b,x) := \begin{cases} S(x) & \text{if } b = 1 \text{ and } x \text{ is not a sink;} \\ x & \text{if } b = 0; \\ 0^n & \text{otherwise.} \end{cases}$$

We define the special element $v^* := (0, 0^n)$. Then, we complete the construction of the reduction from End of Line to Well-Balanced Quotient Weak Pigeon. We can easily see that our reduction can be computed in polynomial time. Thus, what remains is to prove that we can effortlessly obtain a solution to the original instance from every solution to the reduced instance.

By definition, there is no matching violation (i.e., third-type solution). For every non-trivial lonely node $(b,x) \in \{0,1\} \times \{0,1\}^n \setminus \{(0,0^n)\}$ which is a second-type solution, the node x is a solution to the original instance of End of Line. Finally, we consider a collision-type solution $(b,x), (b',y) \in \{0,1\} \times \{0,1\}^n$ such that $(b,x) \not\sim_M (b',y)$ and $C(b,x) = C(b',y)$. Since we suppose that S and P compute valid arcs, it holds that two nodes x and y are sink nodes. Hence, we obtain a solution to the original instance.

3.4 On the Complexity of Weak König

To prove Theorem 15, this section constructs two polynomial-time reductions from Well-Balanced Quotient Weak Pigeon to Weak König and the opposite direction.

Lemma 18. *There is a polynomial-time reduction from* Well-Balanced Quotient Weak Pigeon *to* Weak König*.*

Proof. Let $\mathcal{I}_n := \langle C : \{0,1\}^n \to \{0,1\}^{n-1}; M : \{0,1\}^n \to \{0,1\}^n \rangle$ be any instance of Well-Balanced Explicit Quotient Pigeon. We now construct an instance $\mathcal{J}_{n+1} := \langle P : \{0,1\}^{n+1} \to \{0,1\}^{n+1}; S : \{0,1\}^{n+1} \to \{0,1\}^{n+1}; r := 0^{n+1} \rangle$ of Weak König.

Our reduction is inspired by the PPP-hardness proof of König shown in [PPY23]. The function P implicitly defines a graph on 2^{n+1} nodes, where these nodes are labeled with non-negative integers in the range $[0, 2^{n+1} - 1]$. Roughly speaking, the nodes are supposed to be arranged in a binary tree with root 0. We first define the positions of nodes whose binary encodings are in the range $[0, 2^n - 2]$. First, the node 0 is the root, i.e., $P(0) = 0$. Next, consider any node x in $[2^n - 2]$. We define $P(x) := \left\lfloor \frac{x-1}{2} \right\rfloor$ and $S(x) := x + 1$ if x is odd; otherwise, $S(x) := x - 1$. Note that so far, our definition of the functions P and S has absolutely no dependence on the functions C and M.

To finish the definition of the reduced Weak König instance, i.e., the definition of the functions P and S, it remains to consider the nodes whose binary encodings are in the interval $A := [2^n, 2^{n+1} - 1] \cup \{\alpha = 2^n - 1\}$, where the node α is called an additional node. Consider a given node x in this interval such that $x := 2^n + \xi$ for some $0 \leq \xi \leq 2^n - 1$, we let $y := C(\xi) + 2^{n-1} - 1$ and we define $P(x) = y$ and $S(x) := M(\xi) + 2^n$ if $\xi \neq 0$; otherwise $S(x) = \alpha$. Finally, we define $S(\alpha) = 2^n + 0$ and $P(\alpha) = P(2^n + 0)$. Thus, we match the known lonely node with the additional node.

What remains is to prove that we can obtain a solution to the original Well-Balanced Quotient Weak Pigeon instance $\mathcal{I}_n$ from every solution to the reduced Weak König instance $\mathcal{J}_n$. We first observe that there are no first-type and sixth-type solutions; that is, every node x in $[0, 2^{n+1} - 1]$ is reachable to the root node 0 within length n.

Next, we focus on the nodes whose binary encodings are in A. Note that each node in A has a node whose binary encoding is in $[2^{n-1} - 1, 2^n - 2]$ as a parent. Thus, every node is reachable to the root node by applying the function P at most $n + 1$ times. Furthermore, by definition, there is no non-trivial root node.

Consider a node x in A such that $S(x) = x$. This returns a lonely point by the matching function M.

Finally, we consider a collision-type solution defined by the function C. There are 2^{n-1} nodes in the interval $[2^{n-1} - 1, 2^n - 2]$. On the other hand, there are $2^n + 1$ nodes in A. Hence, we have three nodes that have the same parent from the pecking order principle. Such a triple induces a solution to the original instance of WELL-BALANCED QUOTIENT WEAK PIGEON. Thus, we complete constructing a polynomial-time reduction from WELL-BALANCED QUOTIENT WEAK PIGEON to WEAK KÖNIG. □

Lemma 19. *There is a polynomial-time reduction from* WEAK KÖNIG *to* WELL-BALANCED QUOTIENT WEAK PIGEON.

Proof. Let $\mathcal{I}_n := \langle P : \{0,1\}^n \to \{0,1\}^n; S : \{0,1\}^n \to \{0,1\}^n; r \rangle$ be any instance of WEAK KÖNIG. We now construct a WELL-BALANCED QUOTIENT WEAK PIGEON instance $\mathcal{J}_n := \left\langle C : \{0,1\}^n \to \{0,1\}^{n-1}; M : \{0,1\}^n \to \{0,1\}^n; v^* \right\rangle$.

To show the reduction, we construct a graph G with 2^{n-1} nodes, which consists of a binary tree such that the root node has only one child. First, we correspond each string x in $\{0,1\}^n$ to a node on the graph G.

The two functions P and S implicitly describe a graph on 2^n nodes. We wish to reduce it to an instance of WELL-BALANCED EXPLICIT QUOTIENT PIGEON in polynomial time. To do this, we define the circuit $C : \{0,1\}^n \to \{0,1\}^{n-1}$, where the domain of C will be the nodes of the graph defined by P.

For each node $x \in \{0,1\}^n$, we consider the following sequence $a_0, \ldots, a_k$ of distinct nodes in $\{0,1\}^n$ such that $a_0 := x$, $P(a_k) = a_k$, and $a_i := P(a_{i-1})$ for every $i \in [k]$. We say that a node x is a *bad* position if it holds at least one of the following:

1. $a_k \neq r$; that is, the node a_k is a Non-Unique Root witness;
2. $k \geq n$; that is, the x node is a Long Path certificate or a Far Away certificate.

Furthermore, a node x is said to be *good* if x is not a bad position. For every node x in $\{0,1\}^n$, we define the encode function $\mathrm{Enc} : \{0,1\}^n \to \mathbb{Z}_{\geq 0}$ as follows:

$$\mathrm{Enc}(x) := \begin{cases} 0 & \text{if } P(x) = x; \\ \mathrm{Enc}(P(x)) + 2^{\sigma(x)} & \text{otherwise,} \end{cases}$$

where $\sigma(x) = 1$ if x is lexicographically greater than $S(x)$; otherwise, $\sigma(x) = 0$. By definition, it is not hard to see that if a node x is a good position, then $\mathrm{Enc}(P(x)) \leq 2^{n-1} - 2$.

Then, we define the hole function $C : \{0,1\}^n \to \{0,1\}^{n-1}$ as follows: For each node $x \in \{0,1\}^n$,

$$C(x) := \begin{cases} 2^{n-1} - 1 & \text{if } x = r \text{ or } x \text{ is a bad position;} \\ \mathrm{Enc}(P(x)) & \text{otherwise.} \end{cases}$$

The definition of the matching function $M : \{0,1\}^n \to \{0,1\}^n$ is the same as the sibling function S. Therefore, each of the matching violations or the lonely nodes is a solution to the original Weak König instance. To complete the reduction, we need to show that we can obtain a solution to the original Weak König instance $\mathcal{I}_n$ from every collision-type solution to the reduced instance $\mathcal{J}_n$.

Let x and y be two nodes such that $x \not\sim_S y$ and $C(x) = C(y)$; that is, a collision-type solution. Then, we consider the following two sequences:

(a) $a_0, \ldots, a_k$, where $a_0 := x$, $P(a_k) = a_k = r$, $a_i := P(a_{i-1})$ for each $i \in [k]$;

(b) $b_0, \ldots, b_k$, where $b_0 := y$, $P(b_k) = b_k = r$, $b_i := P(b_{i-1})$ for each $i \in [k]$.

Our goal is to show that there is a solution to the original instance $\mathcal{I}_n$ in the above sequences. The discussion is followed by induction. First, we suppose that $a_{k-1} \neq b_{k-1}$. In this case, it satisfies that $a_{k-1} \not\sim_S b_{k-1}$ since $\sigma(a_{k-1}) = \sigma(b_{k-1})$. Hence, the pair of a_{k-1} and b_{k-1} is a solution to $\mathcal{I}_n$ since $P(a_{k-1}) = P(b_{k-1})$ and $a_{k-1} \not\sim_S b_{k-1}$. Next, we assume that $a_{k-\ell} \neq b_{k-\ell}$ and $a_{k-j} = b_{k-j}$ for every $j \leq \ell$. In this case it satisfies that $a_{k-\ell} \not\sim_S b_{k-\ell}$ since $\sigma(a_{k-\ell}) = \sigma(b_{k-\ell})$. Hence, the pair of $a_{k-\ell}$ and $b_{k-\ell}$ is a solution to $\mathcal{I}_n$ since $P(a_{k-\ell}) = P(b_{k-\ell})$ and $a_{k-\ell} \not\sim_S b_{k-\ell}$. Finally, we know that $a_0 \not\sim_S b_0$. Therefore, we obtain a solution to the original Weak König instance $\mathcal{I}_n$ from each collision-type solution for the reduced Well-Balanced Quotient Weak Pigeon instance $\mathcal{J}_n$. □

Acknowledgment

This work was partially supported by JST, ACT-X, Grant Number JPMJAX2101.

References

[Bur06] Joshua Buresh-Oppenheim. "On the TFNP complexity of factoring." In: (2006). URL: https://www.cs.toronto.edu/~bureshop/factor.pdf.

[CDT09] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. "Settling the complexity of computing two-player Nash equilibria." In: *J. ACM* 56.3 (2009), 14:1–14:57. DOI: 10.1145/1516512.1516516.

[DFM22] Argyrios Deligkas, John Fearnley, and Themistoklis Melissourgos. "Pizza Sharing Is PPA-Hard." In: *Thirty-Sixth AAAI Conference on Artificial Intelligence*. AAAI Press, 2022, pp. 4957–4965. DOI: 10.1609/AAAI.V36I5.20426.

[DGP09] Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. "The complexity of computing a Nash equilibrium." In: *Commun. ACM* 52.2 (2009), pp. 89–97. DOI: 10.1145/1461928.1461951.

[FG18] Aris Filos-Ratsikas and Paul W. Goldberg. "Consensus halving is PPA-complete." In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2018, pp. 51–64. DOI: 10.1145/3188745.3188880.

[Fle+24] Noah Fleming, Stefan Grosser, Toniann Pitassi, and Robert Robere. "Black-Box PPP is Not Turing-Closed." In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. ACM, 2024. URL: https://eccc.weizmann.ac.il/report/2024/010.

[GHH23] Paul W. Goldberg, Kasper Høgh, and Alexandros Hollender. "The Frontier of Intractability for EFX with Two Agents." In: *Algorithmic Game Theory - 16th International Symposium*. Vol. 14238. Lecture Notes in Computer Science. Springer, 2023, pp. 290–307. DOI: 10.1007/978-3-031-43254-5_17.

[Göö+22] Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. "Further Collapses in TFNP." In: *37th Computational Complexity Conference*. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 33:1–33:15. DOI: 10.4230/LIPICS.CCC.2022.33.

[HG18] Alexandros Hollender and Paul W. Goldberg. "The Complexity of Multi-source Variants of the End-of-Line Problem, and the Concise Mutilated Chessboard." In: *Electron. Colloquium Comput. Complex.* TR18-120 (2018). ECCC: TR18-120. URL: https://eccc.weizmann.ac.il/report/2018/120.

[HKT24] Pavel Hubácek, Erfan Khaniki, and Neil Thapen. "TFNP Intersections Through the Lens of Feasible Disjunction." In: *15th Innovations in Theoretical Computer Science Conference*. Vol. 287. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, 63:1–63:24. DOI: 10.4230/LIPICS.ITCS.2024.63.

[Hol21] Alexandros Hollender. "The classes PPA-k: Existence from arguments modulo k." In: *Theor. Comput. Sci.* 885 (2021), pp. 15–29. DOI: 10.1016/J.TCS.2021.06.016.

[HV21] Pavel Hubácek and Jan Václavek. "On Search Complexity of Discrete Logarithm." In: *46th International Symposium on Mathematical Foundations of Computer Science*. Vol. 202. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 60:1–60:16. DOI: 10.4230/LIPICS.MFCS.2021.60.

[Ish21] Takashi Ishizuka. "The complexity of the parity argument with potential." In: *Journal of Computer and System Sciences* 120 (2021), pp. 14–41. DOI: 10.1016/J.JCSS.2021.03.004.

[Ish24] Takashi Ishizuka. "Corrigendum: PLS is contained in PLC." In: *Electron. Colloquium Comput. Complex.* TR24-002 (2024). ECCC: TR24-002. URL: https://eccc.weizmann.ac.il/report/2024/002.

[Jai+24] Siddartha Jain, Jaiwei Li, Robert Robere, and Zhiyang Xun. "On Pigeonhole Principles and Ramsey in TFNP." In: *Electron. Colloquium Comput. Complex.* TR24-017 (2024). ECCC: TR24-017. URL: https://eccc.weizmann.ac.il/report/2024/017.

[Jer16] Emil Jerábek. "Integer factoring and modular square roots." In: *Journal of Computer and System Sciences* 82.2 (2016), pp. 380–394. DOI: 10.1016/J.JCSS.2015.08.001.

[JPY88] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. "How Easy is Local Search?" In: *Journal of Computer and System Sciences* 37.1 (1988), pp. 79–100. DOI: 10.1016/0022-0000(88)90046-3.

[MP91] Nimrod Megiddo and Christos H. Papadimitriou. "On Total Functions, Existence Theorems and Computational Complexity." In: *Theoretical Computer Science* 81.2 (1991), pp. 317–324. DOI: 10.1016/0304-3975(91)90200-L.

[Pap94] Christos H. Papadimitriou. "On the Complexity of the Parity Argument and Other Inefficient Proofs of Existence." In: *Journal of Computer and System Sciences* 48.3 (1994), pp. 498–532. DOI: 10.1016/S0022-0000(05)80063-7.

[PPY23] Amol Pasarkar, Christos H. Papadimitriou, and Mihalis Yannakakis. "Extremal Combinatorics, Iterated Pigeonhole Arguments and Generalizations of PPP." In: *14th Innovations in Theoretical Computer Science Conference*. Vol. 251. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 88:1–88:20. DOI: 10.4230/LIPICS.ITCS.2023.88.

[Pud15] Pavel Pudlák. "On the complexity of finding falsifying assignments for Herbrand disjunctions." In: *Arch. Math. Log.* 54.7-8 (2015), pp. 769–783. DOI: 10.1007/s00153-015-0439-6.

[SZZ18] Katerina Sotiraki, Manolis Zampetakis, and Giorgos Zirdelis. "PPP-Completeness with Connections to Cryptography." In: *59th IEEE Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 2018, pp. 148–158. DOI: 10.1109/FOCS.2018.00023.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il