How Hard Is It to Rig a Tournament When Few Players Can Beat or Be Beaten by the Favorite?

Zhonghao Wang, Junqiang Peng, Yuxi Liu, Mingyu Xiao*

University of Electronic Science and Technology of China
zhonghao.w@outlook.com, jqpeng0@foxmail.com, 202211081321@std.uestc.edu.cn, myxiao@uestc.edu.cn

Abstract

In knockout tournaments, players compete in successive rounds, with losers eliminated and winners advancing until a single champion remains. Given a tournament digraph D, which encodes the outcomes of all possible matches, and a designated player $v^* \in V(D)$, the TOURNAMENT FIXING problem (TFP) asks whether the tournament can be scheduled in a way that guarantees v^* emerges as the winner. TFP is known to be NP-hard, but is *fixed-parameter tractable* (FPT) when parameterized by structural measures such as the feedback arc set (fas) or feedback vertex set (fvs) number of the tournament digraph. In this paper, we introduce and study two new structural parameters: the number of players who can defeat v^* (i.e., the in-degree of v^*, denoted by k) and the number of players that v^* can defeat (i.e., the out-degree of v^*, denoted by ℓ). A natural question is that: can TFP be efficiently solved when k or ℓ is small? We answer this question affirmatively by showing that TFP is FPT when parameterized by either the in-degree or out-degree of v^*. Our algorithm for the in-degree parameterization is particularly involved and technically intricate. Notably, the in-degree k can remain small even when other structural parameters, such as fas or fvs, are large. Hence, our results offer a new perspective and significantly broaden the parameterized algorithmic understanding of the TOURNAMENT FIXING problem.

Introduction

Knockout tournaments are among the most widely used competition formats, characterized by a series of elimination rounds (Horen and Riezman 1985; Connolly and Rendleman 2011; Groh et al. 2012). In each round, players are paired into matches; losers are eliminated, and winners advance to the next round. This process repeats until a single overall winner remains. Due to their efficiency and simplicity, knockout tournaments are used in major events such as the FIFA World Cup (Scarf and Yusof 2011) and the NCAA Basketball Tournament (Kvam and Sokol 2006). Beyond sports, they are also used in elections, organizational decision-making, and various game-based settings, where sequential elimination serves as a natural selection mechanism (Kim, Suksompong, and Williams 2017; Stanton and Williams 2011a; Ramanujan and Szeider 2017). These tournaments have attracted significant attention in artificial intelligence (Vu, Altman, and Shoham 2009; Williams 2010), economics, and operations research (Rosen 1985; Mitchell 1983; Laslier 1997). Their structural and strategic properties continue to motivate both theoretical and applied studies.

Suppose that we have a favorite player v^*, a natural question arises: Can the structure of the tournament be arranged to ensure that v^* wins? To formalize this, let N be a set of $n = 2^c$ players (for some $c \in \mathbb{N}$). The tournament structure is represented by a complete (unordered) binary tree T with n leaves. A *seeding* is a bijection $\sigma : N \to \text{leaves}(T)$, assigning each player to a unique leaf. The tournament proceeds in rounds: In each round, players whose leaves share a common parent play a match; winners advance and are assigned to the parent node, and the leaves are removed. This process continues until one node, and thus one player, remains: the tournament champion.

The question of ensuring a win for v^* becomes meaningful when predictive information about match outcomes is available. We model this with a tournament digraph $D = (V = N, A)$, and for every pair $u, v \in V$, either $(u, v) \in A$ or $(v, u) \in A$, indicating that u is expected to defeat v if and only if $(u, v) \in A$. We study the following problem:

> TOURNAMENT FIXING Problem (TFP)
> **Input:** A tournament D and a player $v^* \in V(D)$.
> **Question:** Does there exist a seeding σ for these $n = |V(D)|$ players such that the favorite player v^* wins the resulting knockout tournament?

Previous Work. The problem of strategically manipulating a knockout tournament was first introduced by Vu, Altman, and Shoham (2009), initiating a line of research focused on identifying structural properties of the input tournament graph D that allow a designated player v^* to be made the winner (Kim, Suksompong, and Williams 2017; Kim and Williams 2015; Ramanujan and Szeider 2017; Stanton and Williams 2011b).

The computational complexity of TFP, particularly its NP-hardness, was posed as an open question in several early works (Vu, Altman, and Shoham 2009; Williams 2010; Russell and Van Beek 2011; Lang et al. 2012). This was resolved affirmatively by Aziz et al. (2014), who proved that

*Corresponding author

TFP is indeed NP-hard. Additionally, they provided algorithms solving the problem in time $\mathcal{O}(2.83^n)$ with exponential space, or $4^{n+o(n)}$ with polynomial space, where n is the number of players. This result was later improved by Kim and Williams (2015) to $2^n n^{\mathcal{O}(1)}$ time and space. Subsequently, Gupta et al. (2018b) proposed an algorithm with the same time complexity while using only polynomial space.

Given the algorithmic interest in TFP, researchers have also explored its parameterized complexity. Notably, TFP becomes trivial when the input tournament digraph D is acyclic. This motivates the study of structural parameters such as the *feedback arc set* (fas) number p and the *feedback vertex set* (fvs) number q, which represent the minimum number of arc reversals or vertex deletions needed to make D acyclic. These parameters can be significantly smaller than the total number of players n.

Ramanujan and Szeider (2017) first showed that TFP parameterized by the fas number p is FPT by giving an algorithm with running time $p^{\mathcal{O}(p^2)}n^{\mathcal{O}(1)}$. This running time bound was later improved to $p^{\mathcal{O}(p)}n^{\mathcal{O}(1)}$ by Gupta et al. (2018a). Additionally, Gupta et al. (2019) showed that TFP admits a polynomial kernel with respect to the fas number p. In terms of the fvs number q, Zehavi (2023) showed that TFP is FPT parameterized by p by giving a $q^{\mathcal{O}(q)}n^{\mathcal{O}(1)}$-time algorithm. This result also subsumes the best known algorithm parameterized by p since $q \leq p$.

The Motivation and Parameters. The fas and fvs numbers are structural parameters for TFP that have been extensively studied because the problem becomes trivial when either is zero, i.e., when the tournament is acyclic. This motivates their use as parameters in TFP analysis.

It is not difficult to see that when the favorite v^* lies in no cycle, v^* wins if and only if no player beats v^*. Thus, in this case, TFP can be solved in polynomial time. This leads to two new parameters: *subset fas number* (minimum number of arc reversals to exclude v^* from cycles) and *subset fvs number* (minimum number of vertex deletions to achieve the same). Although we do not obtain results for these, we explore relaxed alternatives: the *in-degree* and *out-degree* of v^* in tournament D (denoted by k and ℓ respectively). Clearly, v^* is also cycle-free when either k or ℓ is zero.

These local parameters measure how many players defeat v^* (in-degree) or are defeated by v^* (out-degree). Unlike global measures (fas/fvs), they offer a player-centric perspective. They are also easy to compute and intuitive to interpret, making them highly promising for practical applications. Note that these parameters have also been investigated in other tournament-related problems (Yang and Guo 2017).

Though unrelated to fas/fvs, in-degree and out-degree both are not smaller than sub fas/fvs numbers. Figure 1 shows their relationships. In this paper, we prove FPT for in/out-degree parameterization, but the parameterized complexity for subset fas/fvs remains open.

Our Contribution. We introduce two natural parameterizations for the TOURNAMENT FIXING problem (TFP): the *out-degree* and *in-degree* of the favorite player v^* in the input tournament D. Our main contributions are as follows.

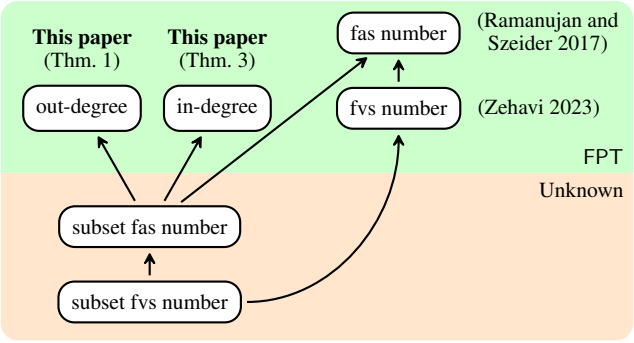

Figure 1: An illustration of the hierarchy of the six parameters, where an arc from parameter x to parameter y denotes $x \leq y$. The green region (upper section) marks parameters for which TFP is proven FPT (including our results), while the orange region (lower section) indicates unresolved cases.

- An FPT algorithm for the out-degree parameterization (Theorem 1) and a lower bound result condition on the Exponential Time Hypothesis (Theorem 2).
- An FPT result for the more challenging in-degree parameterization (Theorem 3).

The out-degree parameterization yields some relatively straightforward results. In contrast, handling the in-degree requires substantial technical innovation and reveals novel structural insights about tournaments. Our approach proceeds in three key steps: (1) **Structural Analysis**: We characterize yes-instances by identifying some structural properties of the tournaments that permit v^*'s victory; (2) **WWF Structure**: We introduce the concept of *winning-witness forests* (WWFs), which is built on the structural properties and handy for our algorithm design; (3) **Algorithmic Reduction**: We develop an FPT algorithm based on the color coding technique, where we reduce TFP to a restricted version of SUBGRAPH ISOMORPHISM problem.

Preliminaries

Notations and tournaments. Let $[n] = \{1, 2, \ldots, n\}$. For a digraph (or a tournament) D, we denote by $V(D)$ its vertex set and by $A(D)$ its arc set. Whenever we consider a *rooted tree*, we treat it as a directed graph where each arc is from the parent to the child. Given a vertex set $X \subseteq V(D)$, we use $D[X]$ to denote the sub-digraph of D induced by the vertices in X. If graph F satisfies $V(F) \subseteq V(D)$ and $A(F) \subseteq A(D)$, we call it a subgraph of D and say $F \subseteq D$. We denote by (u, v) an arc from vertex u to vertex v, and say that u is an in-neighbor of v and v is an out-neighbor of u. For a vertex $v \in V(D)$, let $N_{\text{out}}(v)$ and $N_{\text{in}}(v)$ denote the out-neighborhood and in-neighborhood of v in D, respectively. We will always use v^* denote the favorite in the tournament, and let $\ell = |N_{\text{out}}(v^*)|$ and $k = |N_{\text{in}}(v^*)|$.

Recalling the structure of a knockout tournament, when there are $n = 2^c$ players for some $c \in \mathbb{N}$, the competition unfolds over $\log n$ successive rounds. In each round $r \in [\log n]$, a total of $2^{\log n - r}$ matches are played, with each

match involving two players and producing exactly one winner who advances to the next round. For a tournament D, a seeding σ is called a *winning seeding* for v^* if it makes v^* the winner of the resulting knockout tournament.

Given a tournament D and a seeding σ, we also use the following notion to describe the actual matches played in each round in the resulting knockout tournament.

Definition 1 (match set and sequence)**.** *Let D be a tournament with n players. Fixing a seeding σ, we define:*

- *For each round $r \in [\log n]$, the set of matches that occurred in round r is denoted by $M_r \subsetneq A(D)$, where $|M_r| = 2^{\log n - r}$. Each pair $(u, v) \in M_r$ represents player u beating player v in round r. Let $V(M_r)$ denote the set of players who participated in some match in M_r;*
- *A* match set sequence *is an ordered collection of match sets over all rounds, defined as $\mathcal{M} = \{M_1, \ldots, M_{\log n}\}$.*

We say that a match set sequence $\{M_1, \ldots, M_{\log n}\}$ is valid *for D if the following conditions hold:*

- *If $(u, v) \in M_r$ for some $r \in [\log n]$, then $(u, v) \in A(D)$;*
- *$V(M_1) = V(D)$, and for every round $r \in [\log n - 1]$, the set of winners in M_r is exactly $V(M_{r+1})$.*

Given a seeding σ, it induces a unique match set sequence, denoted by $\mathcal{M}(\sigma)$. In contrast, if a match set sequence $\mathcal{M}^*$ is valid, then there exists at least one seeding σ^* such that $\mathcal{M}(\sigma^*) = \mathcal{M}^*$.

To further characterize the outcome of each round in terms of winners and losers, we use the following notations: Let D be a tournament with n players, and let σ be a seeding, and $\{M_1, \ldots, M_{\log n}\}$ be the corresponding match set sequence. For each round $r \in [\log n]$, let $C_r(\sigma) = \{u \mid (u, v) \in M_r\}$ denote the set of players remaining in the tournament after round r. By convention, we extend the domain to include $r = 0$ and define $C_0(\sigma) = V(D)$, which represents the initial set of all players before any matches have been played. Similarly, for each $r \in [\log n]$, we define the set of players eliminated in round r as $L_r(\sigma) = \{v \mid (u, v) \in M_r\}$. If the seeding σ is clear from context, we may omit it and simply write C_r and L_r.

Binomial Arborescence. Given a rooted forest T and a vertex $v \in V(T)$, let T_v denote the subtree of T rooted of v, consisting of v and all its descendants in T. An *arborescence* is a rooted directed tree such that all arcs are directed away from the root. In order to reformulate TFP, we introduce the concept of a *binomial arborescence* (see Fig. 2).

Definition 2 (unlabeled binomial arborescence)**.** *An* unlabeled binomial arborescence (UBA) *T is defined recursively as follows (see (Williams 2010) also):*

- *A single node v is a UBA rooted at v.*
- *Given two vertex-disjoint UBAs of equal size, T_u rooted at u and T_v rooted at v, adding a directed arc from u to v yields a new UBA rooted at u.*

Let D be a directed graph. If T is a subgraph of D with $V(T) = V(D)$, then T is called a labeled spanning binomial arborescence (LBA) *of D.*

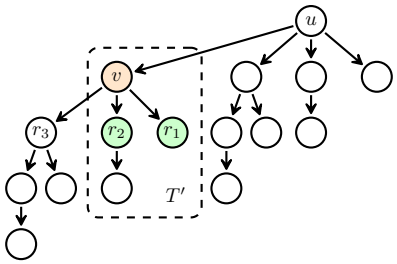

Figure 2: T_u is a UBA with 16 vertices, and the subtree T' formed by T_{v_1}, T_{v_2} and v is a UBA with $2^2 = 4$ vertices.

For any integer $j \in \mathbb{N}$, there exists a unique UBA with 2^j vertices. We give some simple properties of UBAs.

Observation 1. *Let T be a UBA with 2^p vertices and $v \in V(T)$, then:*

- *The subtree T_v of T is itself a UBA;*
- *Suppose v has q children $r_1, \ldots, r_q$. Then, these children can be ordered so that the number of children of each r_i is exactly $i - 1$, and their subtrees $T_{r_1}, \ldots, T_{r_q}$ contains $2^0, 2^1, \ldots, 2^{q-1}$ vertices, respectively;*
- *For any $s \in [q]$, taking v as the root and attaching only the subtrees $T_{r_1}, \ldots, T_{r_s}$ as children. The resulting tree T' is a UBA of size 2^s.*

A seeding of $V(D)$ uniquely corresponds to an LBA of D. The concept of LBA plays a central role in our formulation due to the following proposition, which establishes a direct connection between LBAs and TFP.

Proposition 1 (Williams (2010))**.** *Let D be a tournament with $v^* \in V(D)$. There is a seeding of these players in D such that v^* wins the resulting knockout tournament if and only if D admits an LBA rooted at v^*.*

Proposition 1 reduces TFP to the problem of finding an LBA rooted at the designated winner v^* in the given tournament D. We will adopt this perspective to solve the problem.

Graph Isomorphism. For two digraphs F and D, we call they are *isomorphic* if there exists a bijective function $f : V(F) \to V(D)$, called an *isomorphism*, such that for every pair of vertices $u, v \in V(F)$, $(u, v) \in A(F)$ if and only if $(f(u), f(v)) \in A(D)$. We will utilize the following efficient parameterized algorithm in terms of subgraph isomorphism.

Lemma 1 (Gupta et al. (2018b))**.** *Let F and D be directed graphs on n_F and n vertices, respectively, and suppose that the treewidth of the underlying undirected graph of F is* tw. *Let $c : V(D) \to [n_F]$ be a vertex-coloring of D using n_F colors (not necessarily proper). Given two distinguished vertices $f \in V(F)$ and $d \in V(D)$, there exists an algorithm that decides whether D contains a colorful subgraph isomorphic to F, where the isomorphism maps f to d, in time $2^{n_F} \cdot n^{\mathrm{tw}+\mathcal{O}(1)}$ using polynomial space. We denote this algorithm by $\texttt{alg-subgraph}(F, D, f, d, c)$. Moreover, $\texttt{alg-subgraph}$ can also return a copy of F in D (if one exists) in the same running time bound. This algorithm can immediately solve TFP in $2^n \cdot n^{\mathcal{O}(1)}$ time, where n is the number of players, denoted by $\texttt{alg-TFP-exact}(D, v^*)$.*

Parameterized by the Out-degree

We begin with the out-degree of the favorite player v^* as the parameter. This case admits a relatively simple analysis. We first show that TFP is FPT under this parameterization.

Theorem 1. *TFP can be solved in $2^{2^\ell} \cdot n^{\mathcal{O}(1)}$ time, where n is the number of vertices of the input tournament D and ℓ is the out-degree of the favorite player in D.*

Proof. In a knockout tournament, the winner must win $\log n$ matches. Therefore, if $\ell < \log n$, we can safely report that the input instance is a no-instance. Otherwise, if $\ell \geq \log n$, we have $n \leq 2^\ell$. Applying the $2^n \cdot n^{\mathcal{O}(1)}$-time algorithm in (Gupta et al. 2018b) gives us the desired running time. □

We next present a complementary lower bound result under the Exponential Time Hypothesis (ETH) (Impagliazzo, Paturi, and Zane 2001), which states that no 3-SAT algorithm runs in $2^{o(N)}$ time on all N-variable instances.

Theorem 2. *Under ETH, no algorithm solves TFP in $2^{2^{\ell/c}} \cdot n^{\mathcal{O}(1)}$ time for any constant $c > 1$, where n is the number of vertices of the input tournament D and ℓ is the out-degree of the favorite player in D.*

Proof. Impagliazzo, Paturi, and Zane (2001) showed that under ETH, no 3-SAT algorithm runs in $2^{o(M)}$ time on all instances with M clauses. We consider a variant of 3-SAT, called 3-SAT-2-ltr, where every literal appears at most twice. There is a polynomial-time reduction from (Tovey 1984, Lemma 2.1) that transforms an instance of 3-SAT with M clauses to an equivalent 3-SAT-2-ltr instance with $N = \mathcal{O}(M)$ variables. Thus, under ETH, no $2^{o(N)}$-time algorithm exists for 3-SAT-2-ltr.

In the NP-completeness proof for TFP in (Aziz et al. 2014, Theorem 1), the authors actually showed that given an instance F of 3-SAT-2-ltr with N variables, in polynomial time one can build an instance of TFP with a distinguished player who can win the knockout tournament under some seeding if and only if F is satisfiable. Crucially, in the resulting instance of TFP, the number of vertices of the tournament is $n \leq 64N$, and the out-degree of the distinguished player in the tournament is $\ell = \log n \leq \log N + 6$ (see also (Kim and Williams 2015, Theorem 3)).

Suppose, for the sake of contradiction, that there exists an algorithm $\mathcal{A}$ that solves TFP in time $2^{2^{\ell/c}} \cdot n^{\mathcal{O}(1)}$ for some constant $c > 1$. Note that $2^{\ell/c} \leq 2^{(\log N+6)/c} = N^{1/c} \cdot 2^{6/c} \in o(N)$ since $c > 1$. Then, with the above reduction due to (Aziz et al. 2014), we can use $\mathcal{A}$ to solve 3-SAT-2-ltr in $2^{o(N)}$ time, contradicting ETH. □

We remark that Theorem 2 suggests that the simple algorithm in Theorem 1 is almost optimal under ETH.

Parameterized by the In-degree

This section constitutes the core technical contribution of our work. We study the complexity of TFP when parameterized by the in-degree of the favorite player v^* in the input tournament D. The main result is the following theorem:

Theorem 3. *For an input tournament D and a favorite player $v^* \in V(D)$, TFP is* FPT *when parameterized by the in-degree of v^* in D.*

We will devote the rest of this section to proving this result. The proof proceeds in several steps. We begin by analyzing structural properties of yes-instances, and derive useful constraints on the structure of the input tournament. We then reduce the TFP instance to a carefully constructed instance of a restricted version of the SUBGRAPH ISOMORPHISM problem. Finally, we design an FPT algorithm based on the color coding technique (Alon, Yuster, and Zwick 1995), which has also been applied to solve the general SUBGRAPH ISOMORPHISM problem (Amini, Fomin, and Saurabh 2012).

In the following analysis, we assume $k < \log n$ (or more strictly, $k \cdot 2^k < n$), where n is the number of vertices of the input tournament and k is the in-degree of the favorite player in it. This assumption does not affect the FPT of our algorithm: when it does not hold, n is already bounded by a function of k, and we can solve the problem using a 2^n-time exact algorithm (Gupta et al. 2018b) in FPT time.

Structural Properties

We first introduce a critical definition, which will be handy to characterize the property of a winning-seeding for v^*.

Definition 3 (nice rounds and nice seedings)**.** *For a tournament D with n players and a given seeding σ, we say that round $r \in [\log n]$ is* nice *if it satisfies either $|L_r(\sigma) \cap N_{\text{in}}(v^*)| > 0$ or $|C_{r-1}(\sigma) \cap N_{\text{in}}(v^*)| = 0$. A seeding σ is* nice *if all rounds r under σ are nice.*

Intuitively, a winning seeding is nice if, in every round, at least one in-neighbor of v^* is eliminated if it exists. We then show that for any yes-instance of TFP with few in-neighbors of v^*, there always exists a nice winning-seeding for v^*.

Lemma 2 (existence of nice winning-seeding)**.** *Let $I = (D, v^*)$ be a yes-instance of TFP, where $|V(D)| = n$ and $k = |N_{\text{in}}(v^*)| < \log n$. Then, there exists a nice winning-seeding σ for v^*.*

Proof. Let σ^* be an arbitrary winning-seeding for v^* in tournament D, and let its corresponding match set sequence be $\mathcal{M}^* = \{M_1^*, \cdots, M_{\log n}^*\}$. We will prove this lemma by constructing a nice seeding from σ^*.

If σ^* is already nice, then the claim holds. Otherwise, let p be the last round that not nice; that is, $|L_p(\sigma^*) \cap N_{\text{in}}(v^*)| = 0$ and $|C_{p-1}(\sigma^*) \cap N_{\text{in}}(v^*)| > 0$. Furthermore, we have that $|L_{p+1}(\sigma^*) \cap N_{\text{in}}(v^*)| \geq 1$.

We now modify the match set sequence after round $p-1$ to "repair" the non-nice round p, without changing *niceness* of all rounds $r \geq p+1$. Let $X = C_{p-1}(\sigma^*)$ be the set of players participating in round p, and let $W = L_p(\sigma^*)$ be the players who were eliminated in round p under σ^*.

Let σ_W be an arbitrary seeding of the subtournament among $D[W]$, and player $w \in W \subseteq N_{\text{out}}(v^*)$ be the winner of this subtournament under this seeding. The corresponding match set sequence is $\mathcal{M}(\sigma_W) = \{M_1', \ldots, M_{\log n-p}'\}$.

Note that match set sequence

$$\left\{M_{p+1}^* \cup M_1', \ldots, M_{\log n}^* \cup M_{\log n-p}', \{(v^*, w)\}\right\},$$

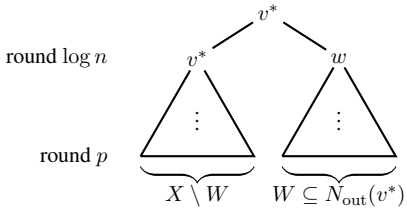

Figure 3: The union of matches among players from $W, X \setminus W$ and a new combined tournament

which is obtained by unioning the match set sequences of W and $X \setminus W$ and adding a new round $\{(v^*, w)\}$, is a valid match set sequence for X (by the structure of knockout tournaments and Definition 1). Then, match set sequence

$$\mathcal{M} = \{M_1^*, \cdots, M_{p-1}^*, \\ M_{p+1}^* \cup M_1', \cdots, M_{\log n}^* \cup M_{\log n - p}', \{(v^*, w)\}\}$$

is also valid for $V(D)$. The corresponding new complete binary tree T of the modified tournament induced by match set sequence $\mathcal{M}$ is shown in Fig 3.

Let σ be a corresponding seeding for $\mathcal{M}$, note that σ maintains that v^* wins, and moreover, the newly modified round p is now nice by construction, since some player from $N_{\text{in}}(v^*) \cap (X \setminus W)$ are eliminated in some match belongs to M_{p+1}^*, and these matches occur in round p now.

If σ is now a nice seeding, the proof is complete. Otherwise, we repeat this process to "repair" the new last non-nice round. Since the index of these rounds strictly decreases in each iteration, the overall iterative procedure must terminate.

Thus, we eventually obtain a nice winning seeding for v^*, as desired, completing the proof. □

For a yes-instance of TFP, the previous lemma guarantees the existence of a nice winning-seeding, in which every round either eliminates at least one player from $N_{\text{in}}(v^*)$ or there is no player from $N_{\text{in}}(v^*)$. This directly implies that all players in $N_{\text{in}}(v^*)$ must be eliminated within the first k rounds, leading to the following corollary.

Corollary 1. *Let $I = (D, v^*)$ be a yes-instance of TFP, where $|V(D)| = n$ and $k = |N_{\text{in}}(v^*)| < \log n$. Then, any nice winning-seeding σ of v^* satisfies $|C_k(\sigma) \cap N_{\text{in}}(v^*)| = 0$, that is, no player from $N_{\text{in}}(v^*)$ lefts after the first k rounds under the seeding σ.*

Building on the previous lemmas and corollary, we now analyze the structural properties of the tournament D in a yes-instance of TFP, which would be useful in the design of our algorithm. We begin with the following lemma, which shows that each in-neighbor $b \in N_{\text{in}}(v^*)$ is contained in a small structured subtree of D.

Lemma 3. *Let $I = (D, v^*)$ be a yes-instance of TFP, where $n = |V(D)|$ and $k = |N_{\text{in}}(v^*)| < \log n$. Let σ^* be a nice winning-seeding for v^* and T be the LBA corresponding to the seeding σ^*. Then, for any vertex $b \in N_{\text{in}}(v^*)$, there exists a directed subgraph F of the LBA T such that:*

- *F is an LBA rooted at some vertex $u \in \{v^*\} \cup N_{\text{out}}(v^*)$ spanning a subset $X \subseteq V(D)$ with $|X| = 2^k$;*
- *If $v^* \in V(F)$, then $u = v^*$;*
- *$b \in V(F)$.*

Proof. Let σ^* be a nice winning-seeding for v^*, whose existence is guaranteed by Lemma 2, and let $\mathcal{M}$ be the corresponding match set sequence. By Proposition 1, there exists an LBA T of D, rooted at v^*, such that for every arc $(u, v) \in A(T)$, it is contained in some match set $M \in \mathcal{M}$.

Fix any $b \in N_{\text{in}}(v^*)$. Starting from b, we traverse upward in T toward the root v^*, stopping at the first vertex u such that the subtree T_u rooted at u satisfies $|V(T_u)| \geq 2^k$. That is, u is the closest ancestor of b in T such that $|V(T_u)| \geq 2^k$. Let v be the child of u such that $b \in V(T_v)$.

Let $r_1, \ldots, r_p$ be the children of u in T. By Observation 1, we can assume that $|V(T_{r_i})| = 2^{i-1}$ for $i \in [p]$. Since $|V(T_u)| \geq 2^k$, we have $p \geq k$. We define F as the subtree rooted at u, containing the subtrees $T_{r_1}, \ldots, T_{r_k}$ and arcs from u to each r_i ($i \in [k]$). By the definition, it holds that $|V(F)| = |\{u\} \cup V(T_{r_1}) \cup \cdots \cup V(T_{r_k})| = 1 + \sum_{i=1}^{k} 2^{i-1} = 2^k$, and F is an LBA by Observation 1.

Since u has at least k children in T, u must have survived at least k rounds under the winning-seeding σ^*. By Corollary 1, all in-neighbors of v^* are eliminated before round k, which implies $u \notin N_{\text{in}}(v^*)$. Hence, $u \in \{v^*\} \cup N_{\text{out}}(v^*)$. Moreover, since v^* is the root of T, it holds that if $v^* \in V(F)$, then $u = v^*$.

Finally, we prove $b \in V(F)$. Suppose, for the sake of contradition, that $b \in V(T_{r_i})$ for some $i > k$. Then, the upward iteration would have terminated earlier at the root of T_{r_i}, since its size is already at least 2^k. Hence, it holds that $b \in V(T_{r_i})$ for some $i \leq k$, which implies $b \in V(F)$. □

Definition 4 (winning-witness forest)**.** *Let D be a tournament and $v^* \in V(D)$. A subgraph $F \subseteq D$ is called a* winning-witness forest (WWF) *of (D, v^*) if the following conditions hold:*

- *F consists of k vertex-disjoint LBAs, each spanning exactly 2^k vertices from $V(D)$ and rooted at some vertex $u_i \in N_{\text{out}}(v^*) \cup \{v^*\}$, $i \in [k]$;*
- *$N_{\text{in}}(v^*) \subseteq V(F)$;*
- *If $v^* \in V(F)$, then $v^* = u_i$ for exactly one $i \in [k]$.*

The concept of a WWF is central to our algorithmic approach. Intuitively, a WWF is a subgraph of D that contains all in-neighbors of v^*.

Lemma 4 (existence of WWF)**.** *Let $I = (D, v^*)$ be an instance of TFP with $n = |V(D)|$ and $k = |N_{\text{in}}(v^*)|$. If $k \cdot 2^k < n$, then I is a yes-instance if and only if there exists a WWF of (D, v^*).*

Proof. We first consider the forward direction. Assume I is a yes-instance. By Proposition 1, there exists an LBA T rooted at v^* spanning $V(D)$. By Lemma 3, for each in-neighbor $b_i \in N_{\text{in}}(v^*)$, there exists an LBA F_i of size 2^k as a subgraph of T that contains b and whose root lies in $N_{\text{out}}(v^*) \cup \{v^*\}$. Let $\mathcal{F} = \{F_i \mid b_i \in N_{\text{in}}(v^*)\}$ denote the collection of these LBAs and let r_i be the root of F_i.

Recall the construction of these LBAs $F_i, F_j \in \mathcal{F}$. We have the following properties: (1) If $r_i = r_j$, then $F_i = F_j$; (2) If $x \in V(F_i)$ and $x \neq r_i$, then $V(T_x) \subseteq V(F_i)$.

We claim that for any two LBAs $F_i, F_j \in \mathcal{F}$, they are either identical or vertex-disjoint. We consider the following two cases: (1) If $r_i = r_j$, then $F_i = F_j$; (2) If $r_i \neq r_j$, assume that $r_i \in V(F_j)$, then $|V(T_{r_i})| < |V(F_j)| = 2^k$. On the other hand, we have $V(F_i) \subseteq V(T_{r_i})$, and in particular $|V(F_i)| = 2^k \leq |V(T_{r_i})|$. This leads to a contradiction. Therefore, $r_i \notin V(F_j)$, and similarly $r_j \notin V(F_i)$. Hence, F_i and F_j are vertex-disjoint. Then, the digraph F where $V(F) = \bigcup_{F_i \in \mathcal{F}} V(F_i)$ and $A(F) = \bigcup_{F_i \in \mathcal{F}} A(F_i)$ forms a forest of at most k vertex-disjoint LBAs. Since some $F_i \in \mathcal{F}$ may be identical, F may contain fewer than k trees. We next complete F by adding additional vertex-disjoint LBAs until it contains exactly k LBAs each with size of 2^k.

As $k \cdot 2^k < n$, there are enough vertices in $V(D) \setminus V(F)$ to construct the remaining LBAs. We repeatedly select disjoint subsets $X \subseteq V(D) \setminus V(F)$ of size 2^k, and for each X, find an LBA F' in $D[X]$. Note that the root of F' lies outside $N_{\text{in}}(v^*)$ since $V(F)$ already contains all of $N_{\text{in}}(v^*)$. We add these LBAs to F until it contains exactly k LBAs. The resulting forest F is a WWF of (D, v^*).

Next, we consider the reverse direction. Assume F is a WWF of (D, v^*), we prove $I = (D, v^*)$ is a yes-instance by constructing an LBA rooted at v^* and spanning $V(D)$. Let $X = V(D) \setminus V(F)$ be the remaining vertices. Since $k \cdot 2^k < n$, we can partition X into exactly $(n/2^k - k)$ disjoint subsets $X_{k+1}, \ldots, X_{n/2^k}$, each of size 2^k. For each $i > k$, construct an arbitrary LBA F_i of size 2^k inside the induced subgraph $D[X_i]$. We now obtain exactly $n/2^k$ LBAs in total (including the original ones in F).

By construction, these LBAs partition all of $V(D)$ and satisfy the following two properties:

(1) All roots lie in $N_{\text{out}}(v^*) \cup \{v^*\}$. For the LBAs in F, this holds by the definition of WWF. For the additional LBAs constructed from the disjoint sets $X_i \subseteq V(D) \setminus V(F)$, since they do not contain all of $N_{\text{in}}(v^*)$, their roots cannot be in $N_{\text{in}}(v^*)$, and are thus in $N_{\text{out}}(v^*) \cup \{v^*\}$;

(2) Exactly one LBA is rooted at v^*. If $v^* \in V(F)$, then by the definition of WWF, it is the root of some F_i. If $v^* \notin V(F)$, then since $N_{\text{in}}(v^*) \subseteq V(F)$, none of the additional sets X_i contains any in-neighbor of v^*, and thus in the LBA formed from the set containing v^*, it must be the root.

We now merge these LBAs iteratively using the standard UBA generating process: at each step, pair LBAs arbitrarily and connect their roots to form larger LBAs (using the arc between these two roots in D). Repeat this for $\log(n/2^k)$ times. In each step, the number of LBAs halves, and their sizes double. Finally, since all roots of these LBAs lie in $N_{\text{out}}(v^*) \cup \{v^*\}$, we obtain a single LBA spanning all of $V(D)$, rooted at v^*. Thus, I is a yes-instance. □

The Algorithm

We present the algorithm (presented in Algorithm 1) that solves TFP. Note that Algorithm 1 is a one-sided error Monte Carlo algorithm with a constant probability of a false negative, and it can be derandomized in a canonical way.

The core idea of our algorithm is to employ color coding technique to determine the existence of a WWF, which is equivalent to check whether the instance is a yes-instance by Lemma 4. To apply this technique, we now introduce the notion of a *colorful* WWF. A WWF is called colorful under a vertex-coloring if all vertices in it are colored with pairwise distinct colors. The following lemma provides a lower bound on the probability that a WWF (if it exists) becomes colorful under the coloring described in Step 1 of Algorithm 1.

Algorithm 1: `alg-TFP-indeg`(D, v^*)

Input: A tournament D and a favorite player $v^* \in V(D)$.
Parameter: The number of players $n = |V(D)|$, and the in-degree of v^* in D, denoted by k.
Output: Does there exist a seeding for $V(D)$ such that the favorite player v^* wins.

1: **if** $k \cdot 2^k \geq n$ **then**
2: **return** `alg-TFP-exact`(D, v^*)
3: **for** $iter = 1$ to $\lceil \mathrm{e}^{k \cdot 2^k - k} \rceil$ **do**
4: ▷ *Step 1. coloring step* ◁
5: Construct a vertex-coloring $c : V(D) \to [k \cdot 2^k]$ of D as follows:
6: $c(b_i) \leftarrow i$ for each $b_i \in N_{\text{in}}(v^*)$, $i \in [k]$
7: $c(v) \leftarrow$ sample uniformly at random from $\{k+1, k+2, \ldots, k \cdot 2^k\}$ for every $v \in N_{\text{out}}(v^*) \cup \{v^*\}$
8: ▷ *Step 2. detecting a colorful WWF* ◁
9: Construct a digraph F' by creating k UBAs $T_1, \ldots, T_k$, each of size 2^k and rooted at $r_1, \ldots, r_k$, adding a root vertex f, and adding arcs (f, r_i) for all $i \in [k]$
10: Construct a digraph D' by removing all arcs (b, v^*) with $b \in N_{\text{in}}(v^*)$ from D, adding a new vertex d, and adding arcs (d, v) for all $v \in N_{\text{out}}(v^*) \cup \{v^*\}$
11: Construct a vertex-coloring $c' : V(D') \to [k \cdot 2^k + 1]$ of D' based on c by adding $c'(d) \leftarrow k \cdot 2^k + 1$
12: **if** `alg-subgraph`(F', D', f, d, c') **then**
13: **return true**
14: **return false**

Lemma 5. *Let $X \subseteq V(D)$ be a vertex subset of D such that $|X| = k \cdot 2^k$ and $N_{\text{in}}(v^*) \subseteq X$, where $k = |N_{\text{in}}(v^*)|$. Let c be a coloring of $V(D)$ by the coloring step of Algorithm 1 (Lines 5–7). Then the probability that X are colored with pairwise distinct colors is at least e^{-t} where $t = k \cdot 2^k - k$.*

Proof. The fixed coloring of the k vertices in $N_{\text{in}}(v^*)$ leaves $t = k \cdot 2^k - k$ uncolored vertices in X and a total of t available colors. Let $n = |V(D)|$. When the remaining $n - k$ vertices are colored uniformly at random using these t colors, the total number of possible colorings is t^{n-k}. The number of favorable outcomes where the t uncolored vertices in X receive pairwise distinct colors is $t! \cdot t^{n-k-t}$. Therefore, the probability that the t uncolored vertices receive distinct colors is $(t! \cdot t^{n-k-t})/(t^{n-k}) = t! \cdot t^{-t} > \mathrm{e}^{-t}$. □

Next, in Step 2, we reduce the problem of detecting a colorful WWF to checking the output of the subroutine `alg-subgraph` on a carefully constructed instance (see Fig. 4), as shown in the following lemma.

Lemma 6. *For a tournament D and a player $v^* \in V(D)$, where $n = |V(D)|$, $k = |N_{\text{in}}(v^*)|$ and $k \cdot 2^k < n$, and $c : V(D) \to [k \cdot 2^k]$ be a vertex-coloring of D with $k \cdot 2^k$*

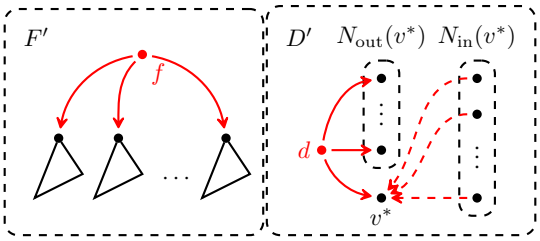

Figure 4: The construction of F' and D'.

colors such that all vertices in $N_{\text{in}}(v^*)$ are assigned pairwise distinct colors that are different from the colors of all other vertices in $V(D)$. Let F' and D', f and d, and c' be the digraphs, vertices, and vertex-coloring constructed in Lines 9-11 of Algorithm 1, respectively. Then, there exists a colorful WWF of (D, v^*) under vertex-coloring c if and only if `alg-subgraph`(F', D', f, d, c') *returns true.*

Proof. We first consider the forward direction. Assume that there exists a colorful WWF H of (D, v^*) under coloring c, consisting of k vertex-disjoint LBAs, and each with the size of 2^k. Let F be a forest composed of k UBAs and each has 2^k vertices, then F is isomorphic to H by Definition 4.

We consider the digraph D'. Vertex d is connected to all roots of the LBAs in H, as they all lie in $N_{\text{out}}(v^*) \cup \{v^*\}$. Moreover, no arc within H is removed, as v^* only can appear as a root in any LBA of H. Therefore, digraph $H' = D'[V(H) \cup \{d\}]$ is simply H augmented with a new root d, which is connected to all the roots of its components.

On the other side, construct F' by adding a vertex f to F and arcs from f to the roots of its LBAs. Then F' is still isomorphic to H' with the isomorphism mapping f to d.

Now consider the coloring c' constructed as in the algorithm. Since H is a colorful WWF under c, the augmented graph H' is also colorful under c' by construction. Therefore, `alg-subgraph`(F', D', f, d, c') returns true.

Next, we consider the reverse direction. Assume that for some vertex-coloring c' of digraph D' constructed in the algorithm, we have `alg-subgraph`(F', D', f, d, c') returns true. Then, there exists a subgraph H' of D' such that F' is isomorphic to H', via an isomorphism which maps f to d, and such that all vertices in H' have distinct colors under c'.

Let H denote the subgraph of H' obtained by removing vertex d and all its incident arcs. Similarly, let F be the subgraph of F' obtained by removing f and its incident arcs. By construction, F is a forest of k vertex-disjoint LBAs, and since F' is isomorphic to H' with the isomorphism maps f to d, it follows that F is also isomorphic to H.

It remains to verify that H satisfies the following three defining properties of a WWF: (1) Each LBA has size 2^k and the forest H has k such trees, matching the structure of F; (2) Each $b_i \in N_{\text{in}}(v^*)$ must be contained in $V(H)$. This holds because each b_i was assigned a unique color from $\{1, \ldots, k\}$, and H' must include a vertex of each color to be colorful. Hence all b_i are present in H; (3) Each LBA in F is rooted at some vertex r_i, which under the isomorphism maps to a vertex in D' connected from d. Since d only connects to vertices in $N_{\text{out}}(v^*) \cup \{v^*\}$, the roots of the LBAs in H must lie there as well. Moreover, since there are no arcs pointing to v^* in D', all non-root vertices of F' cannot map to v^*.

Thus, H is a WWF of (D, v^*), completing the proof. □

By combining the structural characterization of WWF, the correctness of our randomized detection algorithm, and the efficiency of its implementation, we are now ready to prove Theorem 3 via the following lemma:

Lemma 7. *Given an instance $I = (D, v^*)$ of TFP, where $n = |V(D)|$ and $k = |N_{\text{in}}(v^*)|$. Algorithm 1 returns yes with a constant probability if I is a yes-instance and returns no otherwise in time $(2\mathrm{e})^t \cdot n^{\mathcal{O}(1)}$, where $t = k \cdot 2^k - k$.*

Proof. If $k \cdot 2^k \geq n$, we invoke the deterministic algorithm `alg-TFP-exact` in time $2^n \cdot n^{\mathcal{O}(1)}$ by Lemma 1. Since $n \leq k \cdot 2^k$, the bound holds. Otherwise, if $k \cdot 2^k < n$, let $t = k \cdot 2^k - k$. For a yes-instance, a WWF exists by Lemma 4. According to Lemma 5, with probability at least e^{-t}, this WWF becomes colorful during the coloring step. Then, in Step 2, the algorithm `alg-subgraph` correctly identifies the existence of such a colorful WWF in time $2^t \cdot n^{\mathcal{O}(1)}$, by Lemmas 6 and 1. Repeating this process e^t times amplifies the success probability to at least $1 - 1/\mathrm{e}$, resulting in a total running time of $(2\mathrm{e})^t \cdot n^{\mathcal{O}(1)}$, as stated in the lemma. For a no-instance, the algorithm clearly returns no within the same time bound, completing the proof. □

To derandomize the algorithm, we only need to derandomize the coloring step, which can be done in a standard way (Naor, Schulman, and Srinivasan 1995). Roughly speaking, the random coloring step can be replaced with a deterministic enumeration on a family of $\mathrm{e}^{t+o(t)} \cdot \log n$ colorings. Each call to `alg-subgraph` still takes time $2^t \cdot n^{\mathcal{O}(1)}$, so the total deterministic time becomes $(2\mathrm{e})^{t+o(t)} \cdot n^{\mathcal{O}(1)}$.

Conclusion

In this work, we introduce two structural parameters for the Tournament Fixing problem (TFP): the out-degree and in-degree of the favored player v^*, and show that TFP is FPT when parameterized by either of these two parameters.

Another two interesting parameters are subset fas/fvs numbers. TFP can be solved in polynomial time when one of them is zero. Notably, both of these two values are bounded above by the out-degree and in-degree of v^*.

However, it remains open whether TFP is FPT when parameterized by the subset fas number or subset fvs number. In fact, we do not even know whether TFP is NP-hard when either of these values is one. Moreover, unlike the standard fas and fvs numbers which are NP-hard to compute, the subset fas/fvs numbers can be computed in polynomial time by finding a minimum cut between v^- and v^+, where v^- and v^+ are split from v^* such that v^- inherits only the outgoing arcs and v^+ inherits only the incoming arcs of v^*.

All of the above make it compelling to determine the complexity of TFP with respect to subset fas/fvs numbers.

Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions that helped improve the quality of this paper. The work is supported by the National Natural Science Foundation of China, under the grants 62372095, 62502078, 62172077, and 62350710215.

References

Alon, N.; Yuster, R.; and Zwick, U. 1995. Color-coding. *Journal of the ACM (JACM)*, 42(4): 844–856.

Amini, O.; Fomin, F. V.; and Saurabh, S. 2012. Counting subgraphs via homomorphisms. *SIAM Journal on Discrete Mathematics*, 26(2): 695–717.

Aziz, H.; Gaspers, S.; Mackenzie, S.; Mattei, N.; Stursberg, P.; and Walsh, T. 2014. Fixing a balanced knockout tournament. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28.

Connolly, R.; and Rendleman, R. 2011. Tournament qualification, seeding and selection efficiency. Technical report, Technical report 2011-96, tuck school of business.

Groh, C.; Moldovanu, B.; Sela, A.; and Sunde, U. 2012. Optimal seedings in elimination tournaments. *Economic Theory*, 49(1): 59–80.

Gupta, S.; Roy, S.; Saurabh, S.; and Zehavi, M. 2018a. When Rigging a Tournament, Let Greediness Blind You. In *IJCAI*, 275–281.

Gupta, S.; Roy, S.; Saurabh, S.; and Zehavi, M. 2018b. Winning a Tournament by Any Means Necessary. In *IJCAI*, 282–288.

Gupta, S.; Saurabh, S.; Sridharan, R.; and Zehavi, M. 2019. On Succinct Encodings for the Tournament Fixing Problem. In *IJCAI*, 322–328.

Horen, J.; and Riezman, R. 1985. Comparing draws for single elimination tournaments. *Operations Research*, 33(2): 249–262.

Impagliazzo, R.; Paturi, R.; and Zane, F. 2001. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4): 512–530.

Kim, M. P.; Suksompong, W.; and Williams, V. V. 2017. Who can win a single-elimination tournament? *SIAM Journal on Discrete Mathematics*, 31(3): 1751–1764.

Kim, M. P.; and Williams, V. V. 2015. Fixing Tournaments for Kings, Chokers, and More. In *IJCAI*, 561–567.

Kvam, P.; and Sokol, J. S. 2006. A logistic regression/-Markov chain model for NCAA basketball. *Naval Research Logistics (NrL)*, 53(8): 788–803.

Lang, J.; Pini, M. S.; Rossi, F.; Salvagnin, D.; Venable, K. B.; and Walsh, T. 2012. Winner determination in voting trees with incomplete preferences and weighted votes. *Autonomous Agents and Multi-Agent Systems*, 25: 130–157.

Laslier, J.-F. 1997. *Tournament solutions and majority voting*, volume 7. Springer.

Mitchell, W. C. 1983. Toward a theory of the rent-seeking society.

Naor, M.; Schulman, L. J.; and Srinivasan, A. 1995. Splitters and near-optimal derandomization. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, 182–191. IEEE.

Ramanujan, M.; and Szeider, S. 2017. Rigging nearly acyclic tournaments is fixed-parameter tractable. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Rosen, S. 1985. Prizes and incentives in elimination tournaments.

Russell, T.; and Van Beek, P. 2011. An empirical study of seeding manipulations and their prevention. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, 350.

Scarf, P. A.; and Yusof, M. M. 2011. A numerical study of tournament structure and seeding policy for the soccer World Cup Finals. *Statistica Neerlandica*, 65(1): 43–57.

Stanton, I.; and Williams, V. V. 2011a. Manipulating single-elimination tournaments in the Braverman-Mossel model. In *Workshop on Social Choice and Artificial Intelligence*, volume 87.

Stanton, I.; and Williams, V. V. 2011b. Rigging tournament brackets for weaker players. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, 357.

Tovey, C. A. 1984. A simplified NP-complete satisfiability problem. *Discrete applied mathematics*, 8(1): 85–89.

Vu, T.; Altman, A.; and Shoham, Y. 2009. On the complexity of schedule control problems for knockout tournaments. In *AAMAS (1)*, 225–232.

Williams, V. 2010. Fixing a tournament. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, 895–900.

Yang, Y.; and Guo, J. 2017. Possible winner problems on partial tournaments: A parameterized study. *Journal of Combinatorial Optimization*, 33(3): 882–896.

Zehavi, M. 2023. Tournament fixing parameterized by feedback vertex set number is FPT. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 5876–5883.