Improved Upper Bounds for Finding Tarski Fixed Points*

XI CHEN, Columbia University, USA
YUHAO LI, Columbia University, USA

We study the query complexity of finding a Tarski fixed point over the k-dimensional grid $\{1,\dots,n\}^k$. Improving on the previous best upper bound of $O(\log^{\lceil 2k/3\rceil} n)$ [7], we give a new algorithm with query complexity $O(\log^{\lceil (k+1)/2\rceil} n)$. This is based on a novel decomposition theorem about a weaker variant of the Tarski fixed point problem, where the input consists of a monotone function $f:[n]^k\to[n]^k$ and a monotone sign function $b:[n]^k\to\{-1,0,1\}$ and the goal is to find a point $x\in[n]^k$ that satisfies *either* $f(x)\le x$ and $b(x)\le 0$ *or* $f(x)\ge x$ and $b(x)\ge 0$.

CCS Concepts: • **Theory of computation** → **Exact and approximate computation of equilibria**; **Design and analysis of algorithms**.

Additional Key Words and Phrases: Query Complexity, Tarski Fixed Points, Supermodular Games

ACM Reference Format:
Xi Chen and Yuhao Li. 2022. Improved Upper Bounds for Finding Tarski Fixed Points. In *Proceedings of the 23rd ACM Conference on Economics and Computation (EC '22), July 11–15, 2022, Boulder, CO, USA*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3490486.3538297

1 INTRODUCTION

In 1955, Tarski [11] proved that every monotone[1] function $f:L\to L$ over a complete lattice $(L,\preceq)$ has a fixed point, i.e. $x\in L$ with $f(x)=x$. Tarski's fixed point theorem has extensive applications in game theory and economics, where it has been used to establish the existence of important solution concepts such as pure equilibria in supermodular games [9, 12, 13]. As a byproduct, search problems for these solution concepts naturally reduce to the problem of finding Tarski fixed points, which motivates the investigation of its computational complexity. More compelling motivations for studying Tarski fixed points came from a recent work of Etessami, Papadimitriou, Rubinstein and Yannakakis [5], where they discovered new connections of the Tarski fixed point problem with well studied complexity classes such as PPAD and PLS, as well as reductions from Condon's (Shapley's) stochastic games [3] to the Tarski fixed point problem. However, our current understanding of the complexity of Tarski fixed points remains rather limited, whether it is about the query complexity of finding a Tarski fixed point or the white box version (i.e., when the function is given as a Boolean circuit) of the problem (e.g., whether the problem is complete in the class CLS [6, 8] as the intersection of PPAD and PLS). This is in sharp contrast with Brouwer's fixed point theorem [1, 2, 10], the other fixed point theorem that played a major role in economics.

In this paper we study the query complexity of finding a Tarski fixed point in the complete lattice $([n]^k,\preceq)$ over the k-dimensional grid $[n]^k=\{1,\dots,n\}^k$ and equipped with the natural partial

*Supported by NSF grants CCF-1563155, CCF-1703925, IIS-1838154, CCF-2106429 and CCF-2107187.
[1]We say f is monotone if $f(a)\preceq f(b)$ whenever $a\preceq b$.

EC '22, July 11–15, 2022, Boulder, CO, USA

ACM ISBN 978-1-4503-9150-4/22/07...$15.00
https://doi.org/10.1145/3490486.3538297

order over $\mathbb{Z}^k$, where $a \preceq b$ if $a_i \le b_i$ for all $i \in [k]$. An algorithm under this model is given n and k and has query access to an unknown monotone function f over $[n]^k$. Each round the algorithm can send a query $x \in [n]^k$ to reveal $f(x)$ and the goal is to find a fixed point of f using as few queries as possible. We will refer to this problem as $\textsc{Tarski}(n, k)$.

Back in 2011, Dang, Qi, and Ye [4] obtained an $O(\log^k n)$-query algorithm for $\textsc{Tarski}(n, k)$ when k is fixed. Their algorithm is based on a natural binary search strategy over coordinates. No progress had been made on the problem until recently. In [5], Etessami et al. showed that (among other results) the upper bound $O(\log^2 n)$ for $\textsc{Tarski}(n, 2)$ [4] is indeed tight (even for randomized algorithms), which suggested that the algorithm of Dang et al. might be optimal for all fixed k. However, surprisingly, Fearnley, Pálvölgyi and Savani [7] recently showed that the algorithm of [4] is not optimal by giving an algorithm for $\textsc{Tarski}(n, k)$ with $O(\log^{\lceil 2k/3 \rceil} n)$ queries.

Our contribution. Our main result is an improved upper bound for the complexity of $\textsc{Tarski}$:

Theorem 1.1. *For any fixed k, there is an* $O\big(\log^{\lceil (k+1)/2 \rceil} n\big)$*-query algorithm for* $\textsc{Tarski}(n, k)$.

Our algorithm is based on a new variant of the Tarski fixed point problem which we refer to as $\textsc{Tarski}^*$. It is inspired by the $O(\log^2 n)$-query algorithm of [7] for $\textsc{Tarski}(n, 3)$ (its inner algorithm in particular). Our main contribution is a novel decomposition theorem for $\textsc{Tarski}^*$, which leads to a more efficient recursive scheme for performing binary search on coordinates of the grid. We discuss the variant $\textsc{Tarski}^*$ and its decomposition theorem next.

1.1 Sketch of the Algorithm

The algorithm of [7] is obtained by combining an $O(\log^2 n)$-query algorithm for $\textsc{Tarski}(n, 3)$ and a decomposition theorem. Their algorithm for $\textsc{Tarski}(n, 3)$ consists of an outer algorithm and an $O(\log n)$-query inner algorithm. Given $f : [n]^3 \to [n]^3$ as the input function, the outer algorithm starts by running the inner algorithm to solve the following problem:

- Find a point $x \in [n]^3$ with $x_3 = \lceil n/2 \rceil$ such that x is either prefixed ($f(x) \preceq x$) or postfixed ($x \preceq f(x)$). Note that even though we focus on a layer of the grid (with $x_3 = \lceil n/2 \rceil$), the condition on x being either prefixed or postfixed is about all three dimensions.

Once such a point x is found, the outer algorithm can shrink the search space significantly by only considering $\mathcal{L}_{x,(n,n,n)}$ if x is postfixed, or $\mathcal{L}_{(1,1,1),x}$ if x is prefixed, where we write $\mathcal{L}_{a,b}$ to denote the grid with points $c : a \preceq c \preceq b$. In both cases we obtain a grid $\mathcal{L}_{a,b}$ such that $a \preceq b$, $a \preceq f(a)$ and $f(b) \preceq b$. These conditions together guarantee that f maps $\mathcal{L}_{a,b}$ to itself and f has a fixed point in $\mathcal{L}_{a,b}$ (see Lemma 2.4). Given that the side length of a dimension goes down by a factor of 2 after each call to the inner algorithm, it takes no more than $O(\log n)$ calls to reduce the search space to a grid $\mathcal{L}_{a,b}$ with $b_i - a_i \le 1$ and then a fixed point can be found by brute force. The query complexity of the overall algorithm of [7] for $\textsc{Tarski}(n, 3)$ is $O(\log^2 n)$.

After obtaining the $O(\log^2 n)$-query algorithm for $\textsc{Tarski}(n, 3)$, [7] uses it to solve higher dimensional $\textsc{Tarski}$ by proving a *decomposition theorem*: if $\textsc{Tarski}(n, a)$ can be solved in $q(n, a)$ queries and $\textsc{Tarski}(n, b)$ can be solved in $q(n, b)$ queries, then $\textsc{Tarski}(n, a + b)$ can be solved in $O(q(n, a) \cdot q(n, b))$ queries. Combined with the $O(\log^2 n)$-query algorithm for $\textsc{Tarski}(n, 3)$, they obtain an $O(\log^{\lceil 2k/3 \rceil} n)$-query algorithm for $\textsc{Tarski}(n, k)$.

Our key idea is to develop a new decomposition theorem directly on the problem solved by the inner algorithm of [7], and only apply the outer algorithm at the very end. More formally we refer to the following problem as $\textsc{Tarski}^*(n, k)$:[2]

[2]Note that our formal definition in Section 3 will look different; the problems they capture are the same though.

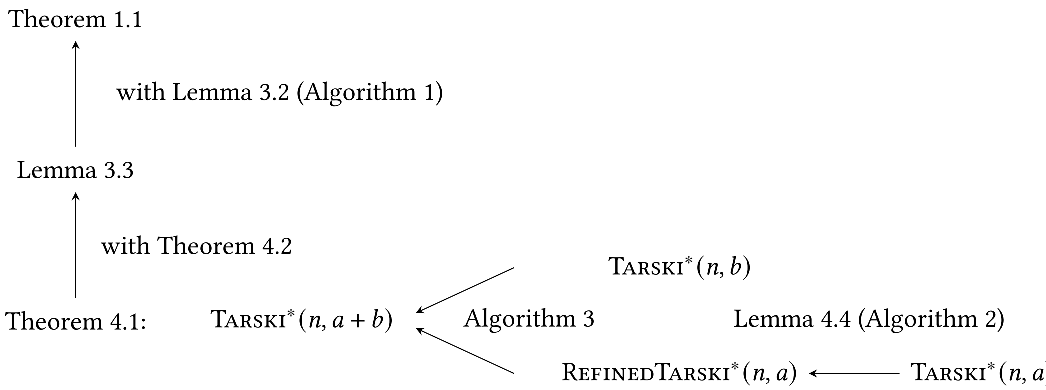

Fig. 1. A Proof Sketch

- Given a monotone function $f : [n]^{k+1} \to [n]^{k+1}$, find a point x with $x_{k+1} = \lceil n/2 \rceil$ such that x is either prefixed or postfixed. As mentioned earlier, the condition on x being either prefixed or postfixed is about all $k+1$ dimensions.

Similar to the outer algorithm of [7], any algorithm for Tarski$^*(n, k)$ can be used as a subroutine to solve Tarski$(n, k+1)$ with an $O(\log n)$-factor overhead (see Lemma 3.2).

The main technical contribution of this work is the proof of a new decomposition theorem for Tarski*: if Tarski$^*(n, a)$ can be solved in $q(n, a)$ queries and Tarski$^*(n, b)$ can be solved in $q(n, b)$ queries, then Tarski$^*(n, a+b)$ can be solved in $O(q(n, a) \cdot q(n, b))$ queries. Now despite sharing the same statement/recursion, the proof of our decomposition theorem requires a number of new technical ingredients compared to that of [7]. This is mainly due to the extra coordinate (i.e., coordinate $k+1$) that appears in Tarski* but not in the original Tarski.

One obstacle is that the solution found by Tarski* appears to be too weak to directly prove the new decomposition theorem. In particular, if one gets a postfixed point $x \preceq f(x)$ as a solution to Tarski$^*(n, k)$, both $x_{k+1} = f(x)_{k+1}$ or $x_{k+1} < f(x)_{k+1}$ could happen, and this uncertainty would cause the proof strategy adopted by [7] to fail. Instead we introduce a stronger variant of Tarski* called RefinedTarski* (see Definition 4.3) which poses further conditions on its solution regarding coordinate $k+1$. Given the same input, RefinedTarski* asks for two points $p^\ell \preceq p^r$ with $p^\ell_{k+1} = p^r_{k+1} = \lceil n/2 \rceil$ such that p^ℓ is postfixed in the first k coordinates, p^r is prefixed in the first k coordinates, and one of the following three conditions hold:

(1) $p^\ell_{k+1} < f(p^\ell)_{k+1}$;
(2) $p^r_{k+1} > f(p^r)_{k+1}$; or
(3) $f(p^\ell)_{k+1} - p^\ell_{k+1} = f(p^r)_{k+1} - p^r_{k+1} = 0$.

While RefinedTarski* looks much stronger than Tarski*, surprisingly we show in Lemma 4.4 that it can be solved by a small number of calls to Tarski*. With RefinedTarski* as the bridge, we are able to prove the new decomposition theorem and obtain the improved bound for Tarski.

2 PRELIMINARIES

We start with the definition of *monotone* functions and state Tarski's fixed point theorem:

Definition 2.1 (Monotone functions). Let $(\mathcal{L}, \preceq)$ be a complete lattice. A function $f : \mathcal{L} \to \mathcal{L}$ is said to be monotone if $f(a) \preceq f(b)$ for all $a, b \in \mathcal{L}$ with $a \preceq b$.

Theorem 2.2 (Tarski). *For any complete lattice $(\mathcal{L}, \preceq)$ and any monotone function $f : \mathcal{L} \to \mathcal{L}$, there must be a point $x_0 \in \mathcal{L}$ such that $f(x_0) = x_0$, i.e., x_0 is a fixed point.*

In this paper we work on the query complexity of Tarski(n, k), i.e., the problem of finding a Tarski fixed point over a k-dimensional grid $([n]^k, \preceq)$, where $[n]$ denotes $\{1, 2, \cdots, n\}$ and $\preceq$ denotes the natural partial order over $\mathbb{Z}^k$: $a \preceq b$ if and only if $a_i \le b_i$ for every $i \in [k]$. For $a, b \in \mathbb{Z}^k$ with $a \preceq b$, we write $\mathcal{L}_{a,b}$ to denote the set of points $x \in \mathbb{Z}^k$ with $a \preceq x \preceq b$. A point $x \in [n]^k$ is called a *prefixed* point of f if $f(x) \preceq x$; a point $x \in [n]^k$ is called a *postfixed* point of f if $x \preceq f(x)$.

Let $S \subseteq \mathbb{Z}^k$ be a finite set of points. A point $p \in \mathbb{Z}^k$ is an *upper bound* of S if $x \preceq p$ for all $x \in S$. We say p is the *least upper bound* of S if p is an upper bound of S and $p \preceq q$ for every upper bound q of S (i.e., $p_i = \max_{x \in S} x_i$ for all $i \in [k]$). Similarly, a point $p \in \mathbb{Z}^k$ is a lower bound of S if $p \preceq x$ for all $x \in S$. We say p is the *greatest lower bound* of S if p is a lower bound of S and $q \preceq p$ for every lower bound q of S (i.e., $p_i = \min_{x \in S} x_i$ for all $i \in [k]$). We write $\mathsf{LUB}(S)$ and $\mathsf{GLB}(S)$ to denote the least upper bound and the greatest lower bound of S, respectively.

We record the following simple fact:

Fact 2.3. *Let finite $S, T \subseteq \mathbb{Z}^k$ be such that $x \preceq y$ for all $x \in S, y \in T$. Then $\mathsf{LUB}(S) \preceq \mathsf{GLB}(T)$.*

We include a proof of the following simple lemma for completeness:

Lemma 2.4. *Let $f : [n]^k \to [n]^k$ be a monotone function. Suppose $\ell, r \in [n]^k$ satisfy $\ell \preceq r$, $\ell \preceq f(\ell)$ and $f(r) \preceq r$. Then f maps $\mathcal{L}_{\ell,r}$ to itself and has a fixed point in $\mathcal{L}_{\ell,r}$.*

Proof. For any $x \in \mathcal{L}_{\ell,r}$, we have from $\ell \preceq x \preceq r$ that

$$\ell \preceq f(\ell) \preceq f(x) \preceq f(r) \preceq r$$

and thus, $f(x) \in \mathcal{L}_{\ell,r}$. The existence of a fixed point in $\mathcal{L}_{\ell,r}$ follows directly from Tarski's fixed point theorem applied on f over $\mathcal{L}_{\ell,r}$. □

3 REDUCTION TO Tarski*

For convenience we focus on Tarski$(n, k+1)$. Our algorithm for Tarski$(n, k+1)$ (see Algorithm 1) over a monotone function $f : [n]^{k+1} \to [n]^{k+1}$ will first set

$$\ell = 1^{k+1} := (1, \ldots, 1) \quad \text{and} \quad r = n^{k+1} := (n, \ldots, n)$$

and then proceed to find a point $x \in [n]^{k+1}$ with $x_{k+1} = \lceil n/2 \rceil$ that is either prefixed ($f(x) \preceq x$) or postfixed ($x \preceq f(x)$). Note that such a point x must exist since by Tarski's fixed point theorem, there must be a point x with $x_{k+1} = \lceil n/2 \rceil$ such that $f(x)_i = x_i$ for all $i \in [k]$ (a fixed point over the slice $x_{k+1} = \lceil n/2 \rceil$), and such a point must be either prefixed or postfixed; on the other hand, it is crucial that the algorithm is not required to find an x with $f(x)_i = x_i$ for all $i \in [k]$ but just an x that is either prefixed or postfixed. After finding x, the algorithm replaces r by x if x is prefixed, or ℓ by x if x is postfixed. It follows from Lemma 2.4 that f remains a monotone function from $\mathcal{L}_{\ell,r}$ to itself but one of the $k+1$ dimensions gets cut by one half. The algorithm recurses on $\mathcal{L}_{\ell,r}$.

The key subproblem is to find such a point x with $x_{k+1} = \lceil n/2 \rceil$ that is either prefixed or postfixed, which we formulate as the following problem called Tarski*(n, k):

*Definition 3.1 (*Tarski*(n, k)).* Given oracle access to a function $g : [n]^k \to \{-1, 0, 1\}^{k+1}$ satisfying

- For all $x \in [n]^k$ and $i \in [k]$, we have $x_i + g(x)_i \in [n]$; and
- For all $x, y \in [n]^k$ with $x \preceq y$, we have $(x, 0) + g(x) \preceq (y, 0) + g(y)$,

find a point $x \in [n]^k$ such that either $g(x)_i \le 0$ for all $i \in [k+1]$ or $g(x)_i \ge 0$ for all $i \in [k+1]$.

Algorithm 1: Algorithm for $\textsc{Tarski}(n, k+1)$ via a reduction to $\textsc{Tarski}^*(n, k)$

Input: Oracle access to a monotone function $f : [n]^{k+1} \to [n]^{k+1}$.
Output: A fixed point $x \in [n]^{k+1}$ of f with $f(x) = x$.

Let $\mathcal{A}$ be an algorithm for $\textsc{Tarski}^*(n, k)$. Let $\ell = 1^{k+1}$ and $r = n^{k+1}$.
while $|r - \ell|_\infty > 2$ **do**
    Pick an $i \in [k+1]$ with $r_i - \ell_i > 2$ and let

$$L = (\ell_1, \cdots, \ell_{i-1}, \ell_{i+1}, \cdots, \ell_{k+1}) \quad \text{and} \quad R = (r_1, \cdots, r_{i-1}, r_{i+1}, \cdots, r_{k+1}).$$

    Define a new function $g : \mathcal{L}_{L,R} \to \{-1, 0, 1\}^{k+1}$ as follows:

$$g(x) := \big(s_1, \cdots, s_{i-1}, s_{i+1}, \cdots, s_{k+1}, s_i\big)$$

    where $s_j := \operatorname{sgn}\Big(f(x')_j - x'_j\Big)$ and $x' = (x_1, \cdots, x_{i-1}, \lceil(\ell_i + r_i)/2\rceil, x_i, \cdots, x_k)$.
    Run algorithm $\mathcal{A}$ on g to find a point $q \in \mathcal{L}_{\ell,r}$ with $q_i = \lceil(\ell_i + r_i)/2\rceil$ that is either prefixed or postfixed; set $r = q$ if q is prefixed and set $\ell = q$ if q is postfixed.
end
Brute-force search $\mathcal{L}_{\ell,r}$ to find a fixed point and return it.

To see the connection between $\textsc{Tarski}^*(n, k)$ and the subproblem described earlier, one can define $g : [n]^k \to \{-1, 0, 1\}^{k+1}$ using $f : [n]^{k+1} \to [n]^{k+1}$ by letting, for each $x \in [n]^k$,

$$g(x)_{k+1} = \operatorname{sgn}\big(f(x, \lceil n/2\rceil)_{k+1} - \lceil n/2\rceil\big) \quad \text{and} \quad g(x)_i = \operatorname{sgn}\big(f(x, \lceil n/2\rceil)_i - x_i\big)$$

for each $i \in [k]$. On the one hand, it is easy to verify that g satisfies both conditions in Definition 3.1 when f is monotone. On the other hand, every $x \in [n]^k$ with $\{-1, 1\} \not\subseteq \bigcup_{i \in [k+1]} \{g(x)_i\}$ must satisfy that $(x, \lceil n/2\rceil)$ is either prefixed or postfixed in f.

The next lemma shows how to use an algorithm for $\textsc{Tarski}^*(n, k)$ to solve $\textsc{Tarski}(n, k+1)$.

Lemma 3.2. *If $\textsc{Tarski}^*(n, k)$ can be solved in $q(n, k)$ queries, then $\textsc{Tarski}(n, k+1)$ can be solved in $O(2^k + k \log n \cdot q(n, k))$ queries.*

Proof. Suppose that $\mathcal{A}$ is an algorithm for $\textsc{Tarski}^*(n, k)$ with $q(n, k)$ queries. We present Algorithm 1 and show that it can solve $\textsc{Tarski}(n, k+1)$ in $O(2^k + k \log n \cdot q(n, k))$ queries.

Correctness. The proof of correctness is based on the observation that $\ell \preceq r$, $\ell \preceq f(\ell)$ and $f(r) \preceq r$ at the beginning of each while loop, which we prove below by induction. The basis is trivial. For the induction step, assume that it holds at the beginning of the current while loop. Then f maps $\mathcal{L}_{\ell,r}$ to itself and thus, g satisfies both conditions in Definition 3.1. As a result, $\mathcal{A}$ can be used to find a point q that is either prefixed or postfixed in f. (Formally, one needs to embed g over $\mathcal{L}_{L,R}$ in the subgrid $\mathcal{L}_{1^k, R-L}$ of $[n]^k$ and define $g' : [n]^k \to \{-1, 0, 1\}^{k+1}$ such that solving $\textsc{Tarski}^*(n, k)$ on g' gives us q.) The way ℓ or r is updated at the end of the loop makes sure that the statement holds at the beginning of the next loop.

The last line of the algorithm makes sure that it returns a fixed point at the end.

Query complexity. Each while loop of Algorithm 1 costs $q(n, k)$ queries. After each loop, the side length of a dimension goes down by a factor of 2. So there are no more than $O(k \log n)$ rounds and thus, the query complexity of Algorithm 1 is $O(2^k + k \log n \cdot q(n, k))$. ◻

We prove the following upper bound for solving $\textsc{Tarski}^*$ in the next section.

Lemma 3.3. *There is an $O(\log^{\lceil k/2\rceil} n)$-query algorithm for $\textsc{Tarski}^*(n, k)$.*

4 A DECOMPOSITION THEOREM FOR TARSKI*

The proof of Lemma 3.3 uses the following decomposition theorem for TARSKI*:

THEOREM 4.1. *If TARSKI*(n, a) can be solved in $q(n, a)$ queries and TARSKI*(n, b) can be solved in $q(n, b)$ queries, then TARSKI*$(n, a + b)$ can be solved in $O((b + 1) \cdot q(n, a) \cdot q(n, b))$ queries.*

We prove Theorem 4.1 in the rest of this section. We also note that the algorithm of [7] can be used to solve the 2-dimensional TARSKI* (see Theorem 14 in [7]), even though they didn't define TARSKI* formally in the paper. This leads to the following theorem about TARSKI*:

THEOREM 4.2 ([7]). *There is an $O(\log n)$-query algorithm for TARSKI*$(n, 2)$.*

Lemma 3.3 follows directly by combining Theorem 4.1 and Theorem 4.2.

4.1 A refined version of TARSKI*

We start the proof of our new decomposition theorem (Theorem 4.1). To this end we first introduce a refined version of TARSKI*.

Definition 4.3 (REFINEDTARSKI(n, k)).* Given a function $g : [n]^k \to \{-1, 0, 1\}^{k+1}$ satisfying

- For all $x \in [n]^k$ and $i \in [k]$, we have $x_i + g(x)_i \in [n]$; and
- For all $x, y \in [n]^k$ with $x \preceq y$, we have $(x, 0) + g(x) \preceq (y, 0) + g(y)$,

find a pair of points $p^\ell, p^r \in [n]^k$ such that $p^\ell \preceq p^r$,

$$g(p^\ell)_t \geq 0 \quad \text{and} \quad g(p^r)_t \leq 0, \quad \text{for all } t \in [k]$$

and one of the following conditions meets

(1) $g(p^\ell)_{k+1} = 1$;
(2) $g(p^r)_{k+1} = -1$; or
(3) $g(p^\ell)_{k+1} = g(p^r)_{k+1} = 0$.

We note that any solution p^ℓ, p^r of REFINEDTARSKI* would imply a solution of corresponding TARSKI* problem directly by returning either p^ℓ or p^r. The following lemma shows that, in fact, these two problems are computationally equivalent in their query complexity.

LEMMA 4.4. *If TARSKI*(n, k) can be solved in $q(n, k)$ queries, then REFINEDTARSKI*(n, k) can be solved in $O(q(n, k))$ queries.*

PROOF. Suppose that $\mathcal{A}$ is an algorithm to solve TARSKI*(n, k) with $q(n, k)$ queries. We show that Algorithm 2 will solve REFINEDTARSKI* in $O(q(n, k))$ queries.

Correctness. It is easy to verify that g^+ over $[n]^k$ satisfies both conditions of Definition 4.3. So the point p^* returned by algorithm $\mathcal{A}$ on line 3 is either a prefixed or a postfixed point of g^+. If $g(p^*)_{k+1} = 1$, then p^ℓ will be updated to p^*. By the definition of TARSKI* we have $g(p^*)_t \geq 0$ for all $t \in [k]$, which means p^ℓ, p^r will meet the first condition of REFINEDTARSKI*. When $g(p^*)_{k+1} = -1$ p^r will be updated to p^* and p^ℓ, p^r will meet the second condition of REFINEDTARSKI*.

Now we can assume $g(p^*)_{k+1} = 0$. So p^ℓ is updated to p^* and p^r remains n^k. Note that p^* is a solution of TARSKI* under g^+ and $g^+(p^*)_{k+1} = 1$ (because $g(p^*)_{k+1} = 0$). So $g^+(p^*)_t \geq 0$ for all $t \in [k + 1]$ and thus, $g(p^\ell)_t \geq 0$ for all $t \in [k]$.

Consider the point q^* returned by algorithm $\mathcal{A}$ on g^- over $\mathcal{L}_{p^\ell,p^r}$ on line 7. If $g(q^*)_{k+1} = 1$, then p^ℓ will be updated to q^* and p^l, p^r will meet the first condition of REFINEDTARSKI*. Otherwise, we have $g(q^*)_{k+1} \leq 0$. Since $p^\ell \preceq q^*$, by the second property of function g, we know $0 = g(p^\ell)_{k+1} \leq g(q^*)_{k+1} \leq 0$, i.e., $g(q^*)_{k+1} = 0$. With the definition of g^-, we know that $g^-(q^*) = -1$. Note that q^* is a solution to TARSKI* on g^-, so we have $g^-(q^*)_t \leq 0$ for all $t \in [k + 1]$. In this case, p^r will be updated as q^*, so p^ℓ, p^r will meet the third condition of REFINEDTARSKI*.

Algorithm 2: Algorithm for RefinedTarski$^*(n,k)$ via a reduction to Tarski$^*(n,k)$

Input: Oracle access to $g : [n]^k \to \{-1,0,1\}^{k+1}$ that satisfies the conditions in Definition 4.3.
Output: A solution to RefinedTarski$^*(n,k)$ on g.

1 Let $\mathcal{A}$ be an algorithm for Tarski$^*(n,k)$. Let $p^\ell = 1^k$ and $p^r = n^k$.
2 Construct a new function $g^+ : [n]^k \to \{-1,0,1\}^k \times \{-1,1\}$ as follows:

$$\begin{cases} g^+(x)_i = g(x)_i, \quad \text{for all } i \in [k] \\ \text{If } g(x)_{k+1} \geq 0, \text{ then } g^+(x)_{k+1} = 1; \text{ if } g(x)_{k+1} = -1, \text{ then } g^+(x)_{k+1} = -1 \end{cases}$$

3 Run algorithm $\mathcal{A}$ to find a solution p^* to Tarski$^*(n,k)$ on g^+ over $[n]^k$.
4 If $g^+(p^*)_{k+1} = 1$, set $p^\ell \leftarrow p^*$; if $g^+(p^*)_{k+1} = -1$, set $p^r \leftarrow p^*$.
5 If $g(p^*)_{k+1} \neq 0$, **return** the pair of points p^ℓ, p^r.
6 Construct a new function $g^- : [n]^k \to \{-1,0,1\}^k \times \{-1,1\}$ as follows:

$$\begin{cases} g^-(x)_i = g(x)_i, \quad \text{for all } i \in [k] \\ \text{If } g(x)_{k+1} \leq 0, \text{ then } g^-(x)_{k+1} = -1; \text{ if } g(x)_{k+1} = 1, \text{ then } g^-(x)_{k+1} = 1 \end{cases}$$

7 Run algorithm $\mathcal{A}$ to find a solution q^* to Tarski$^*(n,k)$ on g^- over $\mathcal{L}_{p^\ell,p^r}$. (This can be done by embedding g^- over $\mathcal{L}_{p^\ell,p^r}$ inside $[n]^k$ and running $\mathcal{A}$.)
8 If $g^-(q^*)_{k+1} = 1$, set $p^\ell \leftarrow q^*$; if $g^-(q^*)_{k+1} = -1$, set $p^r \leftarrow q^*$.
9 **return** the pair of points p^ℓ, p^r.

Query Complexity. Algorithm 2 just calls the algorithm $\mathcal{A}$ at most two times on line 3 and line 7, so the query complexity of Algorithm 2 is $O(q(n,k))$. □

Now we are ready to prove Theorem 4.1.

Proof of Theorem 4.1. Suppose $\mathcal{A}$ is a query algorithm to solve Tarski$^*(n,a)$ in $q(n,a)$ queries and $\mathcal{B}$ is a query algorithm to solve Tarski$^*(n,b)$ in $q(n,b)$ queries. We will show that Algorithm 3 can solve Tarski$^*(n,a+b)$ in $O((b+1)\cdot q(n,a)\cdot q(n,b))$ queries.

Overview. At a high level, Algorithm 3 will run $\mathcal{B}$ for Tarski$^*(n,b)$ on a function $h : [n]^b \to \{-1,0,1\}^{b+1}$ built on the go using the input $g : [n]^{a+b} \to \{-1,0,1\}^{a+b+1}$ of Tarski$^*(n,a+b)$ (that satisfies the conditions in Definition 3.1). Let $q^1,\ldots,q^{i-1} \in [n]^b$ be the $i-1$ queries that $\mathcal{B}$ has made so far, for some $i \geq 1$, and let $r^1,\ldots,r^{i-1} \in \{-1,0,1\}^{b+1}$ be the query results on h. Let $q^i \in [n]^b$ be the new query made by $\mathcal{B}$ in the ith round. Our challenge is to use g (its restriction on points with the last b coordinates being q^i) to come up with an $r^i \in \{-1,0,1\}^{b+1}$ as the answer $h(q^i)$ to the query such that

(1) Lemma 4.7: All results $(q^1,r^1),\ldots,(q^i,r^i)$ are consistent with the conditions of Definition 3.1, i.e., $q^j_i + r^j_i \in [n]$ for all $i \in [b]$ and $(q^j,0)+r^j \preceq (q^{j'},0)+r^{j'}$ for all j,j' with $q^j \preceq q^{j'}$; and
(2) Lemma 4.8: When q^i is a solution to Tarski$^*(n,b+1)$ on h, i.e., either $r^i_t \geq 0$ for all $t \in [b+1]$ or $r^i_t \leq 0$ for all $t \in [b+1]$, we can use q^i to obtain a solution to Tarski$^*(n,a+b)$ on the original input function g.

To obtain r^i, we need to run $\mathcal{A}$ $b+1$ times to obtain a pair of points $p^{(\ell,i)}, p^{(r,i)} \in [n]^a$ and use $g(p^{(\ell,i)},q^i)$ and $g(p^{(r,i)},q^i)$ to determine r^i. A crucial component in the computation of $p^{(\ell,i)}$ and $p^{(r,i)}$ is to initialize the search space using pairs $p^{(\ell,j)}, p^{(r,j)}$, $j \in [i-1]$, from previous rounds.

Correctness. We prove a sequence of lemmas about Algorithm 3:

Algorithm 3: Algorithm for $\textsc{Tarski}^*(n, a+b)$ via $\textsc{Tarski}^*(n, a)$ and $\textsc{Tarski}^*(n, b)$

Input: Oracle access to $g : [n]^{a+b} \to \{-1, 0, 1\}^{a+b+1}$ satisfying conditions in Definition 3.1.
Output: A solution to $\textsc{Tarski}^*(n, a+b)$ on g.

1 Let $\mathcal{A}$ be an algorithm for $\textsc{Tarski}^*(n, a)$ and $\mathcal{B}$ be an algorithm for $\textsc{Tarski}^*(n, b)$.
2 Let $i \leftarrow 1$ be the round number.
3 **do**
4   For each previous round $k \in [i-1]$, let $q^k \in [n]^b$ be the point queried by $\mathcal{B}$ and $r^k \in \{-1, 0, 1\}^{b+1}$ be the answer.
5   Given the sequence $((q^1, r^1), \cdots, (q^{i-1}, r^{i-1}))$, let $q^i \in [n]^b$ be the ith query of $\mathcal{B}$.
6   Set (when $i = 1$, set $p^{(\ell,1)} = 1^a$ and $p^{(r,1)} = n^a$)

$$p^{(\ell,i)} \leftarrow \mathsf{LUB}\left(\left\{p^{(\ell,k)} : k \in [i-1] \text{ and } q^k \preceq q^i\right\}\right) \quad \text{and}$$

$$p^{(r,i)} \leftarrow \mathsf{GLB}\left(\left\{p^{(r,k)} : k \in [i-1] \text{ and } q^i \preceq q^k\right\}\right)$$

7   **for** *each j from $a+1$ to $a+b+1$* **do**
8   Define a new function $g_j : [n]^a \to \{-1, 0, 1\}^{a+1}$ as follows:

$$g_j(x) = \left(g(x, q^i)_1, \cdots, g(x, q^i)_a, g(x, q^i)_j\right), \quad \text{for every } x \in [n]^a.$$

9   Run Algorithm 2 with $\mathcal{A}$ to find a solution $p^{(\ell,*)}, p^{(r,*)}$ to $\textsc{RefinedTarski}^*(n, a)$ on g_j over $\mathcal{L}_{p^{(\ell,i)},p^{(r,i)}}$; set $p^{(\ell,i)} \leftarrow p^{(\ell,*)}$ and $p^{(r,i)} \leftarrow p^{(r,*)}$.
10   **end**
11   Construct $r^i \in \{-1, 0, 1\}^{b+1}$ as the query result to q^i:

$$r^i_{t-a} = g(p^{(\ell,i)}, q^i)_t, \quad \text{for each } t \in [a+b+1] \setminus [a].$$

12   If $r^i_t \geq 0$ for all $t \in [b+1]$, **return** $(p^{(\ell,i)}, q^i)$.
13   If $r^i_t \leq 0$ for all $t \in [b+1]$, **return** $(p^{(r,i)}, q^i)$.
14 **while**;

Lemma 4.5. *At the end of each round i, we have $p^{(\ell,i)} \preceq p^{(r,i)}$ and*

$$g(p^{(\ell,i)}, q^i)_t \geq 0 \quad \textit{and} \quad g(p^{(r,i)}, q^i)_t \leq 0, \quad \textit{for all } t \in [a]. \tag{1}$$

Proof. We start with the base case for the first round. We have $p^{(\ell,1)} = 1^a \preceq n^a = p^{(r,1)}$ at the beginning so Equation (1) holds. It is easy to prove by induction that both $p^{(\ell,1)} \preceq p^{(r,1)}$ and Equation (1) hold at the beginning of each for loop on line 7, g_j over $\mathcal{L}_{p^{(\ell,1)},p^{(r,1)}}$ satisfies conditions of $\textsc{Tarski}^*$ during the for loop and thus, both $p^{(\ell,1)} \preceq p^{(r,1)}$ and Equation (1) hold at the end of the for loop. This shows that both of them hold at the end of the first main loop.

The induction step is similar. Assume that both conditions hold for $p^{(\ell,j)}, p^{(r,j)}$ for $j \in [i-1]$. We start by showing that $p^{(\ell,i)}, p^{(r,i)}$ on line 6 satisfy both $p^{(\ell,i)} \preceq p^{(r,i)}$ and Equation (1).

To prove $p^{(\ell,i)} \preceq p^{(r,i)}$, we make the following observation: $p^{(\ell,j_1)} \preceq p^{(r,j_2)}$ for all $j_1, j_2 \in [i-1]$ with $q^{j_1} \preceq q^{j_2}$. We divide the proof into three cases:

Case 0: $j_1 = j_2 = j$. Trivially follows from $p^{(\ell,j)} \preceq p^{(r,j)}$.
Case 1: $j_1 < j_2$. By the inductive hypothesis, we know $p^{(\ell,j_2)} \preceq p^{(r,j_2)}$. Considering the while loop j_2, by the definition on line 6 and $q^{j_1} \preceq q^{j_2}$, we know $p^{(\ell,j_1)} \preceq p^{(\ell,j_2)}$ before the

loop on line 7. Furthermore, by the updating rule on line 9 , we know that $p^{(\ell,j_2)}$ is monotonically non-decreasing, which means $p^{(\ell,j_1)} \preceq p^{(\ell,j_2)}$ holds after while loop j_2. So we can derive that after while loop j_2, $p^{(\ell,j_1)} \preceq p^{(r,j_2)}$.
Case 2: $j_1 > j_2$. By the inductive hypothesis, we know $p^{(\ell,j_1)} \preceq p^{(r,j_1)}$. Considering the while loop j_1, by the definition on line 6 and $q^{j_1} \preceq q^{j_2}$, we know $p^{(r,j_1)} \preceq p^{(r,j_2)}$ before the loop on line 7. Furthermore, by the updating rule on line 9, we know that $p^{(r,j_1)}$ is monotonically non-increasing, which means $p^{(r,j_1)} \preceq p^{(r,j_2)}$ holds after while loop j_1. So we can derive that after while loop j_1, $p^{(\ell,j_1)} \preceq p^{(r,j_2)}$.

Now we move back to our proof of $p^{(\ell,i)} \preceq p^{(r,i)}$. For every $j_1, j_2 \in [i-1]$ such that $q^{j_1} \preceq q^i \preceq q^{j_2}$, we know that $p^{(\ell,j_1)} \preceq p^{(r,j_2)}$. So the same partial order relation of the least upper bound of $p^{(\ell,j_1)}$ and the greatest lower bound of $p^{(r,j_2)}$ also holds, i.e., $p^{(\ell,i)} \preceq p^{(r,i)}$ before the loop on line 7.

Next we prove Equation (1) on line 6. For each $t \in [a]$, given that $p^{(\ell,i)}$ is the LUB, there must exist $j^* \in [i-1]$ such that

$$q^{j^*} \preceq q^i, \quad p^{(\ell,j^*)} \preceq p^{(\ell,i)} \quad \text{and} \quad p_t^{(\ell,j^*)} = p_t^{(\ell,i)}.$$

Since $(p^{(\ell,j^*)}, q^{j^*}) \preceq (p^{(\ell,i)}, q^i)$, we have

$$p_t^{(\ell,j^*)} + g(p^{(\ell,j^*)}, q^{j^*})_t \le p_t^{(\ell,i)} + g(p^{(\ell,i)}, q^i),$$

which implies $g(p^{(\ell,j^*)}, q^{j^*})_t \le g(p^{(\ell,i)}, q^i)_t$. On the other hand, we have $g(p^{(\ell,j^*)}, q^{j^*})_t \ge 0$ by the inductive hypothesis. So $g(p^{(\ell,i)}, q^i)_t \ge 0$. $g(p^{(r,i)}, q^i)_t \le 0$ can be proved similarly.

Given that both $p^{(\ell,i)} \preceq p^{(r,i)}$ and Equation (1) hold at the beginning of the loop on line 7, the rest of the proof is essentially the same as the proof in the base case. It is easy to prove by induction that both $p^{(\ell,i)} \preceq p^{(r,i)}$ and Equation (1) hold at the beginning of each for loop on line 7, g_j over $\mathcal{L}_{p^{(\ell,i)},p^{(r,i)}}$ satisfies conditions of Tarski* during the for loop and thus, both $p^{(\ell,i)} \preceq p^{(r,i)}$ and Equation (1) hold at the end of the for loop. This shows that both of them hold at the end of the main while loop.

This completes the induction and the proof of the lemma. □

Lemma 4.6. *At the end of every round i, we have $g(p_1, q^i)_t = g(p_2, q^i)_t$ for all $p_1, p_2 \in \mathcal{L}_{p^{(\ell,i)},p^{(r,i)}}$ and all $t \in [a+b+1] \setminus [a]$,*

Proof. Consider the end of round t of the for loop on line 7. If $g(p^{(\ell,*)}, q^i)_t = 1$, then for every $p^{(\ell,*)} \preceq p \preceq p^{(r,*)}$, we have $1 = g(p^{(\ell,*)}, q^i)_t \le g(p, q^i)_t$, i.e., $g(p, q^i)_t = 1$. Similarly, if $g(p^{(r,*)}, q^i)_t = -1$, then we have $g(p, q^i)_t = -1$ for every $p^{(\ell,*)} \preceq p \preceq p^{(r,*)}$. For the last case of $g(p^{(\ell,*)}, q^i)_t = g(p^{(r,*)}, q^i)_t = 0$, for every $p^{(\ell,*)} \preceq p \preceq p^{(r,*)}$, we have $0 = g(p^{(\ell,*)}, q^i)_t \le g(p, q^i)_t \le g(p^{(r,*)}, q^i)_t = 0$, i.e., $g(p, q^i)_t = 0$.

For subsequent round $t+1, t+2, \cdots$ of the for loop, we know that $\mathcal{L}_{p^{(\ell,i)},p^{(r,i)}}$ can only shrink. So the property remains. This finishes the proof of the lemma. □

We are now ready to prove the two lemmas needed for the correctness of Algorithm 3:

Lemma 4.7. *For any two rounds j_1 and j_2, if $q^{j_1} \preceq q^{j_2}$, then $(q^{j_1}, 0) + r^{j_1} \preceq (q^{j_2}, 0) + r^{j_2}$.*

Proof. We consider two cases when $j_1 < j_2$ and when $j_1 > j_2$.

Case 1: $j_1 < j_2$. Considering the round j_2, by the definition on line 6 and $q^{j_1} \preceq q^{j_2}$, we know that $p^{(\ell,j_1)} \preceq p^{(\ell,j_2)}$ before the loop on line 7. In the loop on line 7, when $p^{(\ell,j_2)}$ is updated by $p^{(\ell,*)}$, $p^{(\ell,j_2)}$ is monotonically non-decreasing. So at the end of the loop we still have $p^{(\ell,j_1)} \preceq p^{(\ell,j_2)}$. Given that $(p^{(\ell,j_1)}, q^{j_1}) \preceq (p^{(\ell,j_2)}, q^{j_2})$, we have for every $t \in [a+b] \setminus [a]$:

$$q_{t-a}^{j_1} + r_{t-a}^{j_1} = q_{t-a}^{j_1} + g(p^{(\ell,j_1)}, q^{j_1})_t \le q_{t-a}^{j_2} + g(p^{(\ell,j_2)}, q^{j_2})_t = q_{t-a}^{j_2} + r_{t-a}^{j_2},$$

and $r_{b+1}^{j_1} \le r_{b+1}^{j_2}$.

Case 2: $j_1 > j_2$. Case 2 is analogous to Case 1. Considering the round j_1, by the definition on line 6 and $q^{j_1} \preceq q^{j_2}$, we know that $p^{(r,j_1)} \preceq p^{(r,j_2)}$ before the loop on line 7. In the loop on line 7, when $p^{(r,j_1)}$ is updated by $p^{(r,*)}$, $p^{(r,j_1)}$ is monotonically non-increasing, so that $\forall t \in [a+b+1] \setminus [a]$, $g(p^{(r,j_1)}, q^{j_1})_t$ is non-decreasing. So we have $p^{(r,j_1)} \preceq p^{(r,j_2)}$ at the end of the loop on line 7. Using $(p^{(r,j_1)}, q^{j_1}) \preceq (p^{(r,j_2)}, q^{j_2})$ and Lemma 4.6, we have for every $t \in [a+b] \setminus [a]$:

$$
\begin{aligned}
q_{t-a}^{j_2} + r_{t-a}^{j_2} = q_{t-a}^{j_2} + g(p^{(\ell,j_2)}, q^{j_2})_t &= q_{t-a}^{j_2} + g(p^{(r,j_2)}, q^{j_2})_t \\
&\ge q_{t-a}^{j_1} + g(p^{(r,j_1)}, q^{j_1})_t = q_{t-a}^{j_1} + g(p^{(\ell,j_1)}, q^{j_1})_t = q_{t-a}^{j_1} + r_{t-a}^{j_1},
\end{aligned}
$$

and $r_{b+1}^{j_2} \ge r_{b+1}^{j_1}$.

This completes the proof of the lemma. □

Lemma 4.8. *At the end of each round i, if $r_t^i \ge 0$ for $t \in [b+1]$, then $(p^{(\ell,i)}, q^i)$ is a solution to* Tarski*$^*(n, a+b)$ *on g; if $r_t^i \le 0$ for $t \in [b+1]$, then $(p^{(r,i)}, q^i)$ is a solution to* Tarski*$^*(n, a+b)$ *on g.*

Proof. By Lemma 4.5 we have $g(p^{(\ell,i)}, q^i)_t \ge 0$ and $g(p^{(r,i)}, q^i)_t \le 0$ for all $t \in [a]$. So if $r_t^i \ge 0$ for all $t \in [b+1]$, then $g(p^{(\ell,i)}, q^i)_t = r_{t-a}^i \ge 0$ for all $t \in [a+b+1] \setminus [a]$ and thus, $(p^{(\ell,i)}, q^i)$ is a solution to Tarski$^*(n, a+b)$ on g. Similarly if $r_t^i \le 0$ for all $t \in [b+1]$, then $(p^{(r,i)}, q^i)$ is a solution to Tarski$^*(n, a+b)$ on g. □

Query complexity. For each round of Algorithm 3, Algorithm 2 is called $b+1$ times on line 9 and each call of Algorithm 2 will use $O(q(n,a))$ queries. The outer algorithm $\mathcal{B}$ has no more than $q(n,b)$ rounds, which means the query complexity of Algorithm 3 is $O((b+1) \cdot q(n,a) \cdot q(n,b))$. □

5 DISCUSSION AND OPEN PROBLEMS

While progress has been made on improving the upper bounds for finding Tarski fixed points, the techniques for lower bounds remain limited.

For the black-box (query complexity) model studied in this paper, the key question left open is to close the gap between $\Omega(\log^2 n)$ and $O(\log^{\lceil (k+1)/2 \rceil} n)$. The first gap is from Tarski$(n, 4)$, where the lower bound is $\Omega(\log^2 n)$ and the upper bound is $O(\log^3 n)$. Note that if one could improve the lower bound of Tarski$(n, 4)$ to get a tight bound $\Theta(\log^3 n)$, it would imply that $O(\log^2 n)$ is tight for Tarski$^*(n, 3)$ (while the tight bounds of Tarski$^*(n, 1)$ and Tarski$^*(n, 2)$ are $\Theta(\log n)$). Or even relaxing the goal, is it possible to prove a lower bound $\Omega(\log^3 n)$ for Tarski(n, k) when k is a constant, say, $k = 100$?

With regards to the white-box model, it is known that Tarski is in the intersection of PPAD and PLS [5], and so is in CLS [6] and EOPL [8]. It would also be very interesting to see if Tarski is complete for some computational complexity classes.

ACKNOWLEDGMENTS

We thank anonymous reviewers for helpful comments on an earlier draft.

REFERENCES

[1] Xi Chen and Xiaotie Deng. 2008. Matching algorithmic bounds for finding a Brouwer fixed point. *Journal of the ACM (JACM)* 55, 3 (2008), 1–26.

[2] Xi Chen and Xiaotie Deng. 2009. On the complexity of 2D discrete fixed point problem. *Theoretical Computer Science* 410, 44 (2009), 4448–4456.

[3] Anne Condon. 1992. The complexity of stochastic games. *Information and Computation* 96, 2 (1992), 203–224.

[4] Chuangyin Dang, Qi Qi, and Yinyu Ye. 2011. *Computational models and complexities of Tarski's fixed points.* Technical Report. Stanford University.

[5] Kousha Etessami, Christos H. Papadimitriou, Aviad Rubinstein, and Mihalis Yannakakis. 2020. Tarski's Theorem, Supermodular Games, and the Complexity of Equilibria. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA (LIPIcs, Vol. 151)*, Thomas Vidick (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 18:1–18:19. https://doi.org/10.4230/LIPIcs.ITCS.2020.18

[6] John Fearnley, Paul W Goldberg, Alexandros Hollender, and Rahul Savani. 2021. The complexity of gradient descent: CLS= PPAD ∩ PLS. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. 46–59.

[7] John Fearnley, Dömötör Pálvölgyi, and Rahul Savani. 2020. A faster algorithm for finding Tarski fixed points. *arXiv preprint arXiv:2010.02618* (2020).

[8] Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. 2022. Further Collapses in TFNP. *arXiv preprint arXiv:2202.07761* (2022).

[9] Paul Milgrom and John Roberts. 1990. Rationalizability, learning, and equilibrium in games with strategic complementarities. *Econometrica: Journal of the Econometric Society* (1990), 1255–1277.

[10] Christos H Papadimitriou. 1994. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and system Sciences* 48, 3 (1994), 498–532.

[11] Alfred Tarski. 1955. A lattice-theoretical fixpoint theorem and its applications. *Pacific journal of Mathematics* 5, 2 (1955), 285–309.

[12] Donald M Topkis. 1979. Equilibrium points in nonzero-sum n-person submodular games. *Siam Journal on control and optimization* 17, 6 (1979), 773–787.

[13] Donald M Topkis. 1998. *Supermodularity and Complementarity.* Princeton University Press.