Tarski Fixed Points in Quasi-FPT Queries

Xi Chen
Columbia University
xichen@cs.columbia.edu

Ruiquan Gao
Stanford University
ruiquan@cs.stanford.edu

Yuhao Li
Columbia University
yuhaoli@cs.columbia.edu

Aviad Rubinstein
Stanford University
aviad@cs.stanford.edu

Mihalis Yannakakis
Columbia University
mihalis@cs.columbia.edu

Abstract

We study the query complexity of finding a Tarski fixed point over $[n]^k$. Previous work has left a large gap between $\Omega(\log^2 n)$ and $\log^{O(k)} n$. We show that both of the previous upper and lower bounds were far from tight: for every $k \geq 3$,

$$\Omega\left(\left(\frac{\log n}{\mathsf{polylog}(k)}\right)^{\frac{1}{2}\lceil \log k \rceil}\right) \leq \textsc{Tarski}(n,k) \leq O\left(5^k (\log n)^{\lceil \log k \rceil}\right).$$

Succinctly, up to the fixed-parameter factor of 5^k, the complexity is settled at $(\log n)^{\Theta(\log k)}$. In particular, we obtain the first super-polynomial query lower bound for this problem.

1 Introduction

We begin by recalling the definition of monotone functions and Tarski's fixed point theorem.

Definition 1 (Monotone functions)**.** *Let $(\mathcal{L}, \preceq)$ be a complete lattice. A function $f : \mathcal{L} \to \mathcal{L}$ is* monotone *if $f(x) \preceq f(y)$ for all $x, y \in \mathcal{L}$ with $x \preceq y$.*

Theorem 1 ([Tar55])**.** *Let $(\mathcal{L}, \preceq)$ be a complete lattice and $f : \mathcal{L} \to \mathcal{L}$ be a monotone function. Then the set of* fixed points*, i.e., $\{x \in \mathcal{L} : f(x) = x\}$, is a nonempty complete lattice under the order of $\mathcal{L}$. In particular, f has a least fixed point and a greatest fixed point.*

Tarski's fixed point theorem has had a lasting influence on several fields such as mathematics, economics, logic, and computer science. Applications to set theory and topology already appear in Tarski's original paper [Tar55]. In economics, the theorem provides an elegant foundation for analyzing systemic risk in financial networks [EN01]. In logic, it underlies the semantics of fixed point logics such as the modal μ-calculus [Koz83]. In computer science, it provides foundations for the semantics of recursive programs [SS71] and for static program analysis through abstract interpretation [CC77].

In this paper, we study the problem of finding a Tarski fixed point over the k-dimensional grid lattice $([n]^k, \preceq)$, where $[n]^k := \{1, \ldots, n\}^k$ and $x \preceq y$ if $x_i \leq y_i$ for all $i \in [k]$. This setting captures a broad range of computational problems. In game theory, these include finding pure Nash equilibria in supermodular games, an important class of games with strategic complementarities [Top79, Top98, MR90]. In verification, the problems of solving parity games [EJ91, CJK+17], mean-payoff games [ZP96], and Condon's simple stochastic games [Con92] all reduce to finding a Tarski fixed point [EPRY20]. These games play a central role in verification and synthesis; for example, solving parity games is equivalent to model checking the modal μ-calculus [EJS93]. The Tarski problem also subsumes Arrival, which asks whether a train following deterministic switching rules on a directed graph reaches its destination [DGK+17, GHH21].

For supermodular games, Etessami, Papadimitriou, Rubinstein, and Yannakakis [EPRY20] showed that finding a pure Nash equilibrium is computationally equivalent to finding a Tarski fixed point over the grid lattice. Parity games, mean-payoff games, simple stochastic games, and Arrival are also notable for their distinct complexity status: they are among the few natural problems in $\mathsf{NP} \cap \mathsf{coNP}$, yet have resisted polynomial-time algorithms [Wig19]. Tarski fixed points provide a unified framework for several longstanding open problems, and a polynomial-time algorithm for the Tarski problem would therefore resolve all of them at once.

We study the *query complexity* of finding a Tarski fixed point over $[n]^k$. In the black-box query model, an algorithm is given n and k and oracle access to an unknown monotone function $f : [n]^k \to [n]^k$. In each round, the algorithm chooses a query point $x \in [n]^k$ and receives the entire vector $f(x)$. Queries may depend on previous answers, and the algorithm may perform arbitrarily expensive computation between queries. The goal is to find a fixed point of f using as few queries as possible. We use $\textsc{Tarski}(n, k)$ to denote both this problem and its deterministic query complexity, i.e., the minimum worst-case number of queries needed to solve it. We also consider (Las Vegas) randomized algorithms, which are required to always find a fixed point for every monotone input function f. Their query complexity is the worst-case expected number of queries.

The query complexity of Tarski fixed points over $[n]^k$ has attracted significant attention, yet remains poorly understood. On the lower bound side, Etessami, Papadimitriou, Rubinstein, and Yannakakis [EPRY20] proved an $\Omega(\log^2 n)$ lower bound already in two dimensions, even for randomized algorithms. This was generalized by Brânzei, Phillips, and Recker [BPR25] to k dimensions, obtaining an $\tilde{\Omega}(k \log^2 n)$ lower bound. On the upper bound side, Tarski's algorithm

(value iteration) uses $O(nk)$ queries. The recursive binary search algorithm of Dang, Qi, and Ye [DQY11] gives the baseline bound of $O(\log^k n)$ queries. Surprisingly, Fearnley, Pálvölgyi, and Savani [FPS22] improved this bound to $O(\log^2 n)$ in three dimensions. Haslebacher and Lill [HL26] subsequently gave a different algorithm achieving the same bound. More recently, Chen, Li, and Yannakakis [CLY26] obtained an $O(\log^2 n)$-query algorithm in four dimensions. These algorithms achieve optimal query upper bounds in three and four dimensions but are tailored to the geometry of these low-dimensional settings. Their only known extensions to higher dimensions rely on decomposition theorems [FPS22, CL22], which still yield bounds of $\log^{O(k)} n$. Finally the quantum query complexity of the Tarski fixed point problem has also recently attracted attention [Phi26, LMYZ26].

Our contributions. We significantly improve the best known upper and lower bounds for Tarski fixed points, showing that the upper bound of $\log^{O(k)} n$ and the lower bound of $\Omega(\log^2 n)$ were both far from tight.

On the upper-bound side, we give the first algorithm whose query complexity has only logarithmic dependence on k in the exponent of $\log n$.[1]

Theorem 2. *For $k \geq 3$, there is a deterministic algorithm for* $\textsc{Tarski}(n, k)$ *with query complexity*

$$O\left(5^k (\log n)^{1+\lceil \log((k+2)/3) \rceil}\right) = O\left(5^k (\log n)^{\lceil \log k \rceil}\right).$$

Our algorithm builds on the safe function framework developed earlier in [CLY23, CLY26] but is otherwise self-contained. In particular, our algorithm recovers the optimal $O(\log^2 n)$-query bounds in three and four dimensions within a unified framework that applies to arbitrary dimensions, without relying on geometric properties specific to these low-dimensional settings. We present our algorithm in Section 2. The safe function framework is reviewed in Section 2.1 and then we give an overview of the main ideas behind the new algorithm in Section 2.2.

On the lower bound side, we establish the first super-polynomial query lower bound for Tarski fixed points. Specifically, we prove a lower bound of $(\log n)^{\Omega(\log k)}$ for randomized algorithms, even under the promise that the input function has a unique fixed point.

Theorem 3. *For $k \geq 2$, any randomized algorithm for* $\textsc{Tarski}(n, k)$ *(with probability 1) requires*

$$\Omega\left(\left(\frac{\log n}{\mathsf{polylog}(k)}\right)^{\lfloor \log_4(6k+4) \rfloor}\right)$$

queries in expectation in the worst case, even when the input function has a unique fixed point.

This rules out any (randomized) algorithm with query complexity polynomial in k and $\log n$. In particular, this implies that $\textsc{Tarski} \notin \mathrm{FP}^{\mathrm{dt}}$ in the decision-tree model of $\mathsf{TFNP}^{\mathrm{dt}}$ [GHJ+24]. This is in sharp contrast to contraction fixed points under the ℓ_p-norm, which admit polynomial-query algorithms for all $p \in [1, \infty) \cup \{\infty\}$ [CLY24, HLSW25]. This is also the first super-polynomial lower bound for Tarski fixed points on *any* family of finite lattices in the model where the lattice and its order are fully known and only the monotone function is accessed through an oracle.

Remark. OpenAI announced a deterministic quasipolynomial-time algorithm for simple stochastic games and mean-payoff games on October 6, 2026 [Ope26]. In this context, our quasipolynomial query lower bounds suggest that obtaining polynomial-time algorithms for these games requires exploiting additional structure beyond monotonicity alone.

[1] Throughout the paper, log denotes the base-two logarithm.

2 The Algorithm

We prove Theorem 2 for TARSKI in this section. We first recall definitions of two related problems TARSKI* and SAFETARSKI* in Section 2.1. We state our main theorem for SAFETARSKI* in Section 2.2 (Theorem 5), from which Theorem 2 follows. After an overview of the algorithm behind Theorem 5, we develop the new ingredients in Sections 2.3 to 2.7 and then combine them to finish the proof in Section 2.8.

2.1 Preliminaries

The TARSKI* problem. TARSKI is closely related to a problem called TARSKI* [CL22]. The input to TARSKI*(n,k) is a *monotone* function $f:[n]^k \to [n]^k \times \{\pm 1\}$, meaning that $f(x) \preceq f(y)$ for any $x \preceq y$. We will refer to $f(x)_{k+1} \in \{\pm 1\}$ as the *label* of x. Note that monotonicity holds for all $k+1$ coordinates, including the labels. We recall the definition of TARSKI* as follows:

Definition 2 (TARSKI*(n,k))**.** *Given oracle access to a monotone function $f:[n]^k \to [n]^k \times \{\pm 1\}$, find a point $x \in [n]^k$ such that either $(x,0) \preceq f(x)$ or $f(x) \preceq (x,0)$. We also write* TARSKI*(n,k) *to denote its deterministic query complexity. We set* TARSKI*$(n,0) = 1$ *by convention.*

A point $x \in [n]^k$ is called a *positive solution* to f if $(x,0) \preceq f(x)$, and a *negative solution* to f if $f(x) \preceq (x,0)$. We call x a *solution* to f if it is either a positive or a negative solution. The goal of TARSKI* is then to find a solution to the unknown monotone function f.

The standard binary search algorithm can reduce TARSKI(n,k) to TARSKI*$(n,k-1)$ and give the following bound [FPS22, CL22]. See [CLY26] for a sketch of the reduction.

Lemma 1 (Folklore)**.** *For every $k \geq 1$, we have*

$$\text{TARSKI}(n,k) = O\big((\log n) \cdot \text{TARSKI}^*(n,k-1)\big).$$

Previous work improved the query complexity of TARSKI by developing better algorithms for TARSKI*. In particular, it was shown that TARSKI*$(n,2) = O(\log n)$ [FPS22] and TARSKI*$(n,3) = O(\log n)$ [CLY26]. When combined with Lemma 1, these bounds yield optimal $O(\log^2 n)$-query algorithms for TARSKI$(n,3)$ and TARSKI$(n,4)$, respectively.

The SAFETARSKI* problem. The upper bound of TARSKI*$(n,3) = O(\log n)$ was proved using an approach based on *safe* functions, which are monotone functions that satisfy an additional *safety* condition. We need the following notation to define safe functions. Here a *slice* of $[n]^k$ is specified by an $s \in ([n] \cup \{*\})^k$. We write

$$\mathcal{F}(s) = \big\{i \in [k] : s_i = *\big\} \quad \text{and} \quad \mathcal{L}_s = \big\{x \in [n]^k : x_i = s_i \text{ for } i \in [k] \setminus \mathcal{F}(s)\big\}$$

to denote its set of free coordinates and its set of points, respectively. A point $x \in \mathcal{L}_s$ is a fixed point of f on slice s if $f(x)_i = x_i$ for all $i \in \mathcal{F}(s)$; $x \in [n]^k$ is a fixed point of f if $f(x)_i = x_i$ for all $i \in [k]$.

Definition 3 (Safe Functions)**.** *A monotone function $f:[n]^k \to [n]^k \times \{\pm 1\}$ is said to be* safe *if it satisfies the following* safety *condition: f has a* unique *fixed point on* every slice s *of* $[n]^k$.

The approach based on safe functions uses the following SAFETARSKI*(n,k) problem, which is the same as TARSKI*(n,k), except that f is not only monotone but also promised to be safe:

Definition 4 (SafeTarski$^*(n,k)$)**.** *Given oracle access to a safe function $f : [n]^k \to [n]^k \times \{\pm 1\}$, find a point $x \in [n]^k$ such that x is a solution to f. We also write* SafeTarski$^*(n,k)$ *to denote its deterministic query complexity, and set* SafeTarski$^*(n,0) = 1$ *by convention.*

It was shown in [CLY23, CLY26] that SafeTarski* and Tarski* have exactly the same query complexity so it suffices to give algorithms for SafeTarski*:

Theorem 4 ([CLY23, CLY26])**.** *For every $n, k \geq 1$,* Tarski$^*(n,k) =$ SafeTarski$^*(n,k)$.

2.2 Main theorem and proof overview

The main theorem of this section is a recursive algorithm for SafeTarski$^*(n,k)$ that leads to the following recurrence:

Theorem 5. *For every $k \geq 2$, we have*

$$\text{SafeTarski}^*(n,k) = O\Big(\big(k\,2^k \log n\big) \cdot \text{SafeTarski}^*\big(n, \lfloor k/2 \rfloor - 1\big)\Big).$$

Solving the recurrence in Theorem 5 and applying Lemma 1 and Theorem 4, we obtain Theorem 2. We defer the detailed calculation to Appendix A.

Proof overview. To prove Theorem 5, we give an algorithm for SafeTarski$^*(n,k)$ in the rest of this section. We first give an overview of the main technical ideas behind it.

Let Q be the set of queries made so far by an algorithm on SafeTarski*, together with their answers. Given such a query-answer set Q, the unknown function can be any safe function that is consistent with Q. In Section 2.3, we prove Lemma 3, which gives lower and upper bounds on the unique fixed point in every slice s, for any safe function consistent with Q. Using these bounds, we give an explicit definition of a set called $\texttt{Fix}(Q) \subseteq [n]^k$ and show that it always contains the unique fixed point of every safe function consistent with Q. We then incorporate the queried labels (i.e., information about coordinate $k+1$ in Q) to define two subsets, $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$, of $\texttt{Fix}(Q)$, consisting of points in $\texttt{Fix}(Q)$ that can still receive labels -1 and $+1$, respectively.

A key technical lemma (Lemma 9) proved in Section 2.5 shows that if $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$ are *coordinate-separable* along one of the first k coordinates, say coordinate k, in the sense that

$$\max_{x \in \texttt{Fix}^-(Q)} x_k \leq \min_{y \in \texttt{Fix}^+(Q)} y_k,$$

then we are very close to solving SafeTarski$^*(n,k)$: a recursive call to SafeTarski$^*(n,k-1)$ is enough to finish the job. Inspired by Lemma 9, the main goal of our algorithm is to force such a coordinate separation between $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$ using few queries.

For this purpose we measure our progress toward this goal using the number of *bad point-pairs* (x,y): $x \in \texttt{Fix}^-(Q)$ and $y \in \texttt{Fix}^+(Q)$ such that $x_k > y_k$. These are exactly the pairs obstructing coordinate separation along coordinate k. We therefore seek to query-efficiently reduce the number of bad point-pairs. This is done in Sections 2.6 and 2.7: the former establishes the needed structural property of $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$, while the latter gives an algorithmic procedure that exploits it.

More specifically, we prove Lemma 10 in Section 2.6, showing that there always exists a *heavy antipodal set-pair* (S^-, S^+) with $S^- \subseteq \texttt{Fix}^-(Q)$ and $S^+ \subseteq \texttt{Fix}^+(Q)$ satisfying:

1. The set-pair is *heavy*: Eliminating either S^- from $\texttt{Fix}^-(Q)$ or S^+ from $\texttt{Fix}^+(Q)$ would remove at least a $1/2^k$ fraction of bad point-pairs[2]; and

[2]While we call these sets heavy, this $1/2^k$ factor is responsible for the 2^k loss in the recurrence of Theorem 5.

2. The set-pair is *antipodal*: There exists a threshold $t \in [n]^k$ such that, in every coordinate $i \in [k-1]$, all points in S^- and all points in S^+ lie on opposite sides of t_i (i.e., either all points in S^- are at most t_i and all points in S^+ are at least t_i, or vice versa), while in coordinate k, all points in S^+ are at most t_k and all points in S^- are at least $t_k + 1$. (See item (ii) of Lemma 10 for the formal condition.)

In Section 2.7 we prove Lemma 12, which shows that for any such antipodal set-pair (S^-, S^+), one can eliminate at least one of the two sets using a recursive call to SafeTarski$^*(n, \ell)$ for some $\ell \le \lfloor k/2 \rfloor - 1$. Combining Lemma 10 and Lemma 12, we can eliminate at least a $1/2^k$ fraction of the bad point-pairs using one recursive call to SafeTarski$^*(n, \ell)$. Repeating this process eventually eliminates all bad point-pairs, after which $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$ are coordinate-separable along coordinate k, and Lemma 9 can be applied. Finally we combine these ingredients in Section 2.8 to give an algorithm for SafeTarski* that proves Theorem 5.

2.3 Query-answer sets on safe functions

A set $Q = \{(x^1, r^1), \ldots, (x^q, r^q)\}$, for some $q \ge 0$, is called a *query-answer set* if $x^1, \ldots, x^q$ are distinct points in $[n]^k$ and $r^i \in [n]^k \times \{\pm 1\}$ for each $i \in [q]$. We use a query-answer set Q to record queries that an algorithm has made on SafeTarski$^*(n, k)$ during its first q rounds, together with their answers from the oracle. (So $Q = \varnothing$ at the beginning.) We say a query-answer set Q and a function $h : [n]^k \to [n]^k \times \{\pm 1\}$ are *consistent*, denoted by $h \Rightarrow Q$, if $h(x) = r$ for all $(x, r) \in Q$.

Given that the unknown function in SafeTarski$^*(n, k)$ is promised to be safe, we will only be interested in query-answer sets Q that are consistent with at least one safe function h. We refer to such a Q as a *safe* query-answer set. Given a slice s of $[n]^k$, our goal in this subsection is to use information in Q to obtain lower and upper bounds on the unique fixed point on slice s of any safe function $h \Rightarrow Q$ (Lemma 3). These bounds will be the starting point for defining our candidate fixed points $\texttt{Fix}(Q)$ in the next subsection. We note that this subsection is all about the first k coordinates; label information of coordinate $k+1$ in Q will only be used later in the next subsection when we define the two subsets $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$ of $\texttt{Fix}(Q)$.

We start with some definitions. We say a point $y \in \mathcal{L}_s$ is *postfixed* for Q on slice s if $h(y)_i \ge y_i$ for every $i \in \mathcal{F}(s)$ and every monotone[3] function h consistent with Q. Similarly, a point $y \in \mathcal{L}_s$ is *prefixed* for Q on s if $h(y)_i \le y_i$ for every $i \in \mathcal{F}(s)$ and every monotone function h consistent with Q. When $s = (*, \ldots, *)$, we simply say that y is postfixed or prefixed for Q.

We prove the following lemma on the sets of postfixed and prefixed points:

Lemma 2. *Let Q be a safe query-answer set and s be a slice of $[n]^k$. The set of postfixed (prefixed) points for Q on slice s is nonempty and closed under coordinatewise joins (or meets, respectively).*

Proof. Let $I = \mathcal{F}(s)$, and let $\ell, u \in \mathcal{L}_s$ be the bottom and top points of the slice, respectively. That is, for every $i \in [k]$, $\ell_i = 1$ if $i \in I$ and $\ell_i = s_i$ if $i \notin I$, and $u_i = n$ if $i \in I$ and $u_i = s_i$ if $i \notin I$.

For every monotone function $h \Rightarrow Q$, the boundary conditions imply $h(\ell)_i \ge \ell_i$ and $h(u)_i \le u_i$ for every $i \in I$. So ℓ is postfixed and u is prefixed for Q on s, proving that both sets are nonempty.

Next, let $a, b \in \mathcal{L}_s$ be postfixed for Q on s, and set $c = a \vee b$, where $c_i = \max\{a_i, b_i\}$ for every $i \in [k]$. Since a and b agree in all fixed coordinates, $c \in \mathcal{L}_s$. Fix any monotone function h consistent

[3] We use monotone rather than safe functions here for simplicity. Restricting to safe functions can only tighten the lower and upper bounds in Lemma 3, but it is not clear how to exploit this to improve the query complexity.

with Q and any $i \in I$. By interchanging a and b if necessary, we assume without loss of generality that $c_i = a_i$. As $a \preceq c$, monotonicity of h gives

$$h(c)_i \geq h(a)_i \geq a_i = c_i,$$

where the second inequality follows because a is postfixed for Q on s. This shows that c is postfixed for Q on s and thus, the set of postfixed points for Q on s is closed under coordinatewise joins. It can be proved similarly that the set of prefixed points is closed under coordinate meets. □

Given Lemma 2, we know that there is a *greatest* postfixed point for Q on slice s, denoted by $J_s(Q)$, and there is a *least* prefixed point, denoted by $M_s(Q)$. For the all-$*$ slice, we denote these two points as $J(Q)$ and $M(Q)$ for convenience.

The next lemma shows that the unique fixed point x^* on slice s of any safe function consistent with Q must lie between $J_s(Q)$ and $M_s(Q)$:

Lemma 3. *Let Q be a safe query-answer set, and h be any safe function consistent with Q. Then for any slice s of $[n]^k$, the unique fixed point x^* of h on slice s satisfies $J_s(Q) \preceq x^* \preceq M_s(Q)$. This in particular implies that $J_s(Q) \preceq M_s(Q)$.*

Proof. Fix a slice s, and let $I = \mathcal{F}(s)$. Define a map $f_s : \mathcal{L}_s \to \mathcal{L}_s$ by setting

$$f_s(x)_i = \begin{cases} h(x)_i & i \in I \\ s_i & i \notin I \end{cases}, \quad \text{for every } x \in \mathcal{L}_s \text{ and } i \in [k].$$

It is easy to verify that f_s maps $\mathcal{L}_s$ to $\mathcal{L}_s$ and is monotone, i.e., $f_s(x) \preceq f_s(y)$ for any $x \preceq y \in \mathcal{L}_s$. Moreover, a point $x \in \mathcal{L}_s$ is a fixed point of f_s if and only if $h(x)_i = x_i$ for every $i \in I$. Since h is safe, x^* is the unique fixed point of f_s.

To prove the lower bound $J_s(Q) \preceq x^*$, let $x^0 = J_s(Q)$ and define $x^{t+1} = f_s(x^t)$ for $t = 0, 1, \ldots$. Since x^0 is postfixed for Q on s and h is consistent with Q, we have $h(x^0)_i \geq x^0_i$ for all $i \in I$ and thus, $x^0 \preceq f_s(x^0) = x^1$. By an induction, it is easy to show that the sequence of points $x^0, x^1, \ldots$ is nondecreasing. Since all its points belong to the finite set $\mathcal{L}_s$, there is an integer $T \geq 0$ such that $x^T = x^{T+1} = f_s(x^T)$. Hence x^T is a fixed point of f_s, so $x^T = x^*$ by uniqueness. We conclude that $J_s(Q) = x^0 \preceq x^T = x^*$. The upper bound $x^* \preceq M_s(Q)$ can be proved similarly. □

Before moving on, we record the following simple fact that, as an algorithm makes more queries and extends the query-answer set Q, $J_s(Q)$ can only increase and $M_s(Q)$ can only decrease:

Lemma 4. *Let $Q \subseteq Q'$ be two safe query-answer sets. Then $J_s(Q) \preceq J_s(Q')$ and $M_s(Q') \preceq M_s(Q)$ for every slice s of $[n]^k$.*

Proof. Fix a slice s. Given that any monotone function consistent with Q' must also be consistent with Q, the set of postfixed points for Q on s is a subset of that for Q' on s. It then follows that $J_s(Q) \preceq J_s(Q')$. The proof for prefixed points is similar. □

2.4 Candidate fixed points

Let Q be a safe query-answer set. Given Lemma 3, it is natural to define the following set

$$\texttt{Fix}(Q) = \left\{ x \in [n]^k : J_s(Q) \preceq x \preceq M_s(Q) \text{ for every slice } s \text{ with } x \in \mathcal{L}_s \right\}.$$

For any safe function $g \Rightarrow Q$, let x^* be its unique fixed point in $[n]^k$. Then x^* must be the unique fixed point on every slice s with $x^* \in \mathcal{L}_s$. It follows from Lemma 3 that $x^* \in \texttt{Fix}(Q)$. This is why we will refer to $\texttt{Fix}(Q)$ as our set of *candidate fixed points* given Q, since it contains the unique fixed point of every safe function $g \Rightarrow Q$.

The next two lemmas give more properties about $\texttt{Fix}(Q)$. First we show that $\texttt{Fix}(Q)$ always contains $J(Q)$ and $M(Q)$.

Lemma 5. *For any safe query-answer set Q, we have $J(Q) \in \texttt{Fix}(Q)$ and $M(Q) \in \texttt{Fix}(Q)$.*

Proof. Let $a = J(Q)$, and fix any slice s with $a \in \mathcal{L}_s$. We will show that $a = b := J_s(Q)$.

On the one hand, since a is postfixed for Q on the whole grid, it is also postfixed on s. Thus we have $a \preceq b$. On the other hand, to see $b \preceq a$, fix any monotone function $h \Rightarrow Q$. For each $i \in \mathcal{F}(s)$, the definition of b gives $h(b)_i \geq b_i$. For each $i \notin \mathcal{F}(s)$, we have $b_i = a_i$, and monotonicity gives

$$h(b)_i \geq h(a)_i \geq a_i = b_i.$$

So b is postfixed for Q on the whole grid. By the definition of $J(Q)$, we have $b \preceq a$, so $J_s(Q) = a$.

By Lemma 3, we now have $J_s(Q) = a \preceq M_s(Q)$. Since this holds for every slice s with $a \in \mathcal{L}_s$, it follows from definition that $a \in \texttt{Fix}(Q)$. The argument for $M(Q)$ is symmetric. □

The next lemma shows that any point x, at which Q forces a strict inequality of $h(x)_i > x_i$ or $h(x)_i < x_i$ for any $i \in [k]$, must be excluded from $\texttt{Fix}(Q)$. The proof uses a one-dimensional slice and the safety condition to obtain a bound that excludes the point:

Lemma 6. *Let Q be a safe query-answer set. If $x \in [n]^k$ and $i \in [k]$ are such that $h(x)_i > x_i$ for every monotone $h \Rightarrow Q$, or such that $h(x)_i < x_i$ for every monotone $h \Rightarrow Q$, then $x \notin \texttt{Fix}(Q)$.*

Proof. Let x and $i \in [k]$ be such that $h(x)_i > x_i$ for every monotone function $h \Rightarrow Q$. Since these functions take values in $[n]^k \times \{\pm 1\}$, we have $x_i < n$. Let s be the one-dimensional slice through x with free coordinate i, and set $v = x + \mathbf{e}_i$. By monotonicity, h must satisfy

$$h(v)_i \geq h(x)_i \geq x_i + 1 = v_i.$$

So v is postfixed for Q on slice s and thus, $v \preceq J_s(Q)$. Since $v_i > x_i$, the condition $J_s(Q) \preceq x$ fails, and hence $x \notin \texttt{Fix}(Q)$. The case when $h(x)_i < x_i$ for every monotone $h \Rightarrow Q$ is symmetric. □

Next we incorporate the label information of Q in coordinate $k+1$ to define subsets $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$. We define them as follows:

$$\texttt{Fix}^-(Q) = \big\{x \in \texttt{Fix}(Q) : \nexists (y, r) \in Q \text{ with } y \preceq x,\ r_{k+1} = +1\big\} \quad \text{and}$$
$$\texttt{Fix}^+(Q) = \big\{x \in \texttt{Fix}(Q) : \nexists (y, r) \in Q \text{ with } x \preceq y,\ r_{k+1} = -1\big\}.$$

So in $\texttt{Fix}^-(Q)$ (or $\texttt{Fix}^+(Q)$) we exclude $x \in \texttt{Fix}(Q)$ that must be labelled $+1$ (or -1, respectively) in any safe $g \Rightarrow Q$. Note that their union is $\texttt{Fix}(Q)$ but they do not form a partition of $\texttt{Fix}(Q)$ in general. It is intuitive to see that, as more queries are made, all three sets can only shrink:

Lemma 7. *If $Q \subseteq Q'$ are safe query-answer sets, then we have*

$$\texttt{Fix}(Q') \subseteq \texttt{Fix}(Q), \quad \texttt{Fix}^-(Q') \subseteq \texttt{Fix}^-(Q) \quad \textit{and} \quad \texttt{Fix}^+(Q') \subseteq \texttt{Fix}^+(Q).$$

Proof. Let $x \in \texttt{Fix}(Q')$. For every slice s containing x, Lemma 4 gives

$$J_s(Q) \preceq J_s(Q') \preceq x \preceq M_s(Q') \preceq M_s(Q).$$

Thus $x \in \texttt{Fix}(Q)$, and we have proved that $\texttt{Fix}(Q') \subseteq \texttt{Fix}(Q)$.

Moreover, since $Q \subseteq Q'$, any queried label in Q that excludes a point from $\texttt{Fix}^-(Q)$ or $\texttt{Fix}^+(Q)$ also excludes it from the corresponding set for Q'. This proves the other two inclusions. □

So as an algorithm makes more queries, all three sets can only shrink. While $\texttt{Fix}(Q)$ can never be empty, the last lemma of this subsection shows that, if one of $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$ becomes empty, a solution to the unknown function has already been revealed by Q:

Lemma 8. *If $\texttt{Fix}^-(Q) = \varnothing$, then $J(Q)$ must be a positive solution to every safe function $g \Rightarrow Q$.*
If $\texttt{Fix}^+(Q) = \varnothing$, then $M(Q)$ must be a negative solution to every safe function $g \Rightarrow Q$.

Proof. Suppose that $\texttt{Fix}^-(Q) = \varnothing$, and let $a = J(Q)$. By Lemma 5, $a \in \texttt{Fix}(Q)$. Given that $a \notin \texttt{Fix}^-(Q)$ (as $\texttt{Fix}^-(Q) = \varnothing$), there exists $(y, r) \in Q$ with $y \preceq a$ and $r_{k+1} = +1$.

Fix any safe function $g \Rightarrow Q$. Monotonicity gives $g(a)_{k+1} = +1$. Since a is postfixed for Q on the whole grid, we have that a is a positive solution to g. The other case is symmetric. □

2.5 Dimension reduction via coordinate separation

Given Lemma 7 and Lemma 8, the high-level strategy of our algorithm for SafeTarski* is to make queries to shrink $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$, aiming to eliminate one of them with as few queries as possible. A key idea behind our new algorithm in this paper is to not just care about individual sizes of the two sets, but to aim at making the two sets *coordinate-separable*: We say a pair of sets (A, B), where $A, B \subseteq [n]^k$, is *coordinate-separable*[4] if

$$\max_{x \in A} x_k \leq \min_{y \in B} y_k.$$

Equivalently this means that there exists a $t \in [n]$ such that $x_k \leq t \leq y_k$ for all $x \in A$ and $y \in B$.

The following lemma shows that if $(\texttt{Fix}^-(Q), \texttt{Fix}^+(Q))$ is coordinate-separable, then indeed we are very close to eliminating one of them completely, which by Lemma 8 will solve SafeTarski*:

Lemma 9. *Let Q be a safe query-answer set such that $(\texttt{Fix}^-(Q), \texttt{Fix}^+(Q))$ is coordinate-separable. There is an algorithm that, given Q and oracle access to any safe function $g \Rightarrow Q$, makes at most $\textsc{SafeTarski}^*(n, k-1) + 1$ additional queries to find a solution to g.*

Proof. If either $\texttt{Fix}^-(Q)$ or $\texttt{Fix}^+(Q)$ is empty, the claim is trivial by Lemma 8. Below we assume that they are nonempty and, for some $t \in [n]$, satisfy

$$\max_{x \in \texttt{Fix}^-(Q)} x_k \leq t \leq \min_{y \in \texttt{Fix}^+(Q)} y_k.$$

Given a safe function $f \Rightarrow Q$, we focus on its $(k-1)$-dimensional slice $s = (*, \ldots, *, t)$ and induce the following function $f' : \mathcal{L}_s \to \mathcal{L}_s \times \{\pm 1\}$: For $i \in [k-1]$, we have $f'(z)_i = f(z)_i$; $f'(z)_k = t$; and its label in coordinate $k+1$ is defined by

$$f'(z)_{k+1} = +1 \quad \Longleftrightarrow \quad f(z)_k > t \quad \text{or} \quad \big[f(z)_k = t \text{ and } f(z)_{k+1} = +1 \big]. \tag{1}$$

[4]Note that the definition uses specifically coordinate k. This is without loss of generality; the same proof works if we replace k by any coordinate $i \in [k]$.

From now on, we view f' as a function from $[n]^{k-1}$ to $[n]^{k-1} \times \{\pm 1\}$, by skipping coordinate k. It can be easily verified that f' must be safe given that f is safe. Hence f' is a valid input function for $\textsc{SafeTarski}^*(n, k-1)$. Furthermore, given the definition of f' from f, oracle access to f' can be simulated by oracle access to f, query by query.

Given Q and oracle access to any safe function $f \Rightarrow Q$, the algorithm proceeds as follows: It runs an algorithm $\textsc{Alg}$ for $\textsc{SafeTarski}^*(n, k-1)$ on the unknown function f' induced by f. Each time when $\textsc{Alg}$ makes a query $z \in [n]^{k-1}$ about f', our algorithm makes a query y about f, where y is obtained from z by inserting t as its coordinate k. After receiving $r = f(y)$ from the oracle, our algorithm updates Q by adding a new pair (y, r), replies to $\textsc{Alg}$'s query (based on the construction of f' from f), and continues. This process stops within $\textsc{SafeTarski}^*(n, k-1)$ rounds; at the end, $\textsc{Alg}$ returns a point $z \in [n]^{k-1}$ as a solution to f'. Let y be its corresponding point in $[n]^k$. Our algorithm makes one additional query on y (if it has never been queried before). This makes sure that at the end, letting $Q' \supseteq Q$ be the updated query-answer set, Q' must contain a pair (y, r) such that $y \in \mathcal{L}_s$ and one of the following two conditions (with four subcases) holds:

1. $r_i \geq y_i$ for every $i \in [k-1]$ and either (1.1) $r_k > t$ or (1.2) $r_k = t$ and $r_{k+1} = +1$; or
2. $r_i \leq y_i$ for every $i \in [k-1]$ and either (2.1) $r_k < t$ or (2.2) $r_k = t$ and $r_{k+1} = -1$.

We show below that, in each of these four subcases, a solution has already been revealed in Q'.

In Case 1.2, it is clear that y must be a positive solution in any safe function $f \Rightarrow Q'$; in Case 2.2, y must be a negative solution in any safe function $f \Rightarrow Q'$.

Consider Case 1.1. Let $y' \in [n]^k$ be obtained from y by increasing coordinate k by one. Since $h(y) = r$ for every monotone function $h \Rightarrow Q'$, monotonicity implies that y' is postfixed for Q' over the whole grid. Every $x \in \texttt{Fix}(Q')$ satisfies $y' \preceq J(Q') \preceq x$, so $x_k \geq t+1$. As every $x \in \texttt{Fix}^-(Q)$ satisfies $x_k \leq t$ and $\texttt{Fix}^-(Q') \subseteq \texttt{Fix}^-(Q)$, we have $\texttt{Fix}^-(Q') = \varnothing$ and by Lemma 8, $J(Q')$ must be a positive solution to every safe function consistent with Q'. The proof of Case 2.1 is symmetric. □

Inspired by Lemma 9, our goal in the rest of this section is to design a query-efficient strategy to force $(\texttt{Fix}^-(Q), \texttt{Fix}^+(Q))$ to become coordinate-separable. A natural way to measure the amount of work to be done is to count the number of *bad point-pairs*. Here we say (x, y) is a bad point-pair if $x \in \texttt{Fix}^-(Q), y \in \texttt{Fix}^+(Q)$ and $x_k > y_k$. We use $\mathcal{R}(Q)$ to denote the set of bad point-pairs. Note that $(\texttt{Fix}^-(Q), \texttt{Fix}^+(Q))$ is coordinate-separable if and only if $\mathcal{R}(Q) = \varnothing$, so bad point-pairs are obstructions to the coordinate separation that we aim for. Given that by Lemma 7, both $\texttt{Fix}^-(Q)$ and $\texttt{Fix}^+(Q)$ only shrink after each query, so does $\mathcal{R}(Q)$. Naturally our next goal is to come up with a strategy to shrink $\mathcal{R}(Q)$ query-efficiently.

2.6 Existence of a heavy antipodal set-pair

We start by proving Lemma 10, which shows the existence of a *heavy antipodal* set-pair (S^-, S^+), where $S^- \subseteq \texttt{Fix}^-(Q)$ and $S^+ \subseteq \texttt{Fix}^+(Q)$. Being *heavy* means that eliminating either one of them would reduce the size of $\mathcal{R}(Q)$ by at least a $1/2^k$ fraction. The structure of the set-pair being antipodal, to be defined below, will be exploited in the next subsection, where we give an algorithm that can eliminate one of the two sets with few queries.

Lemma 10. *Let Q be a safe query-answer set with $\mathcal{R}(Q) \neq \varnothing$. There exists a set-pair (S^-, S^+) with $S^- \subseteq \texttt{Fix}^-(Q)$ and $S^+ \subseteq \texttt{Fix}^+(Q)$ such that the following two conditions hold:*

(i) The set-pair (S^-, S^+) is heavy*: We have*

$$\big|\{(x,y) \in \mathcal{R}(Q) : x \in S^-\}\big|,\ \big|\{(x,y) \in \mathcal{R}(Q) : y \in S^+\}\big| \geq \frac{|\mathcal{R}(Q)|}{2^k}.$$

(ii) The set-pair (S^-, S^+) is antipodal*: There is a partition $[k] = J \dot\cup H$ with $k \in H$, and a vector $t \in [n]^k$ satisfying that, for every $x \in S^-$ and $y \in S^+$, we have*

$$x_J \preceq t_J \preceq y_J \quad \text{and} \quad x_H \succeq t_H \succeq y_H. \tag{2}$$

Moreover, for coordinate k, we have $x_k \geq t_k + 1$ for every $x \in S^-$.

Proof. We first handle coordinate k. Order the pairs $(x,y) \in \mathcal{R}(Q)$ by y_k, and let q be the y_k-value of a middle pair. Define

$$R^- = \big\{(x,y) \in \mathcal{R}(Q) : y_k \geq q\big\} \quad \text{and} \quad R^+ = \big\{(x,y) \in \mathcal{R}(Q) : y_k \leq q\big\}.$$

Both sets contain at least half of the bad pairs because of the median choice of q. Every $(x,y) \in R^-$ satisfies $x_k > y_k \geq q$, while every $(x,y) \in R^+$ satisfies $y_k \leq q$. Set $t_k = q$, $J = \varnothing$, and $H = \{k\}$.

We now restrict R^- and R^+ one coordinate at a time. For each $i \in [k-1]$, order the pairs in the current R^- by x_i and those in the current R^+ by y_i, and let a, b be the respective coordinate values of middle pairs. If $a \leq b$, restrict R^- to pairs with $x_i \leq a$ and R^+ to pairs with $y_i \geq b$, set $t_i = a$, and put i in J. If $a > b$, restrict R^- to pairs with $x_i \geq a$ and R^+ to pairs with $y_i \leq b$, set $t_i = b$, and put i in H. In either case, each set keeps at least half of its pairs, and the required inequalities hold in coordinate i. Further restrictions preserve all inequalities established in earlier coordinates.

After processing all $k-1$ coordinates, we have a partition $[k] = J \dot\cup H$ with $k \in H$, and the final pair of sets satisfy $|R^-| \geq 2^{-k}|\mathcal{R}(Q)|$ and $|R^+| \geq 2^{-k}|\mathcal{R}(Q)|$. Define

$$S^- = \big\{x : (x,y) \in R^- \text{ for some } y\big\} \quad \text{and} \quad S^+ = \big\{y : (x,y) \in R^+ \text{ for some } x\big\}.$$

By construction, these families satisfy (2), and every $x \in S^-$ has $x_k \geq t_k + 1$, proving item (ii). Every pair in R^- has its first endpoint in S^-, and every pair in R^+ has its second endpoint in S^+. Since both pair sets are subsets of $\mathcal{R}(Q)$, their size bounds imply the two bounds in item (i). □

2.7 Eliminating one set from an antipodal set-pair

Now let Q be a query-answer set and let (S^-, S^+), where $S^- \subseteq \texttt{Fix}^-(Q)$ and $S^+ \subseteq \texttt{Fix}^+(Q)$, be a set-pair that is antipodal with respect to $t \in [n]^k$ (in the sense of item (ii) in Lemma 10). In this subsection we give an algorithm that eliminates one of the sets, i.e., either S^- from $\texttt{Fix}^-(Q)$ or S^+ from $\texttt{Fix}^+(Q)$, using few queries.

We first give a simple lemma about $\texttt{Fix}(Q)$:

Lemma 11. *Let Q be a safe query-answer set and $z \in [n]^k$. For any $I \subseteq [k]$, the following hold:*

(i) If $h(z)_i \geq z_i$ for every $i \in I$ in any monotone $h \Rightarrow Q$, then $x \notin \texttt{Fix}(Q)$ for any x such that

$$x_j \geq z_j \quad \text{for every } j \in [k] \setminus I \quad \text{and} \quad x_i < z_i \quad \text{for some } i \in I.$$

(ii) If $h(z)_i \leq z_i$ for every $i \in I$ in any monotone $h \Rightarrow Q$, then $x \notin \texttt{Fix}(Q)$ for any x such that

$$x_j \leq z_j \quad \text{for every } j \in [k] \setminus I \quad \text{and} \quad x_i > z_i \quad \text{for some } i \in I.$$

Proof. For (i), let x be a point that satisfies the specified conditions, and set $w = x \vee z$. We define a slice s by setting $s_i = *$ if $x_i < z_i$ and $s_i = x_i$ if $x_i \geq z_i$. So $\mathcal{F}(s) \neq \varnothing$ and $\mathcal{F}(s) \subseteq I$. We also have $x, w \in \mathcal{L}_s$ and $x \prec w$, and $w_i = z_i$ for every $i \in \mathcal{F}(s)$. By assumption, we have $h(z)_i \geq z_i$ for all $i \in I$ and every monotone function h consistent with Q. As $z \preceq w$ and $w_i = z_i$ for $i \in \mathcal{F}(s)$, monotonicity of h gives $h(w)_i \geq w_i$ for every $i \in \mathcal{F}(s)$. Thus w is postfixed on s, so $x \prec w \preceq J_s(Q)$, which by definition excludes x from $\texttt{Fix}(Q)$. The proof of (ii) is symmetric. □

Now we give the algorithm that eliminates one set from (S^-, S^+):

Lemma 12. *Let $k \geq 2$. Let Q be a safe query-answer set, and $S^- \subseteq \texttt{Fix}^-(Q)$ and $S^+ \subseteq \texttt{Fix}^+(Q)$ be two nonempty sets such that (S^-, S^+) is an antipodal set-pair in the sense of (ii) in Lemma 10. There is an algorithm that, given Q, (S^-, S^+) and oracle access to any safe function $g \Rightarrow Q$, makes at most $\textsc{SafeTarski}^*(n, \lfloor k/2 \rfloor - 1) + 1$ additional queries to g to extend Q to Q' such that either*

$$S^- \cap \texttt{Fix}^-(Q') = \varnothing \quad \text{or} \quad S^+ \cap \texttt{Fix}^+(Q') = \varnothing.$$

Proof. First, consider the special case when $J = \varnothing$. Then we know that $y \preceq t \preceq x$ for every $x \in S^-$ and $y \in S^+$. The algorithm simply queries t and let $Q' = Q \cup \{(t, r)\}$ be the resulting query-answer set. If $r_{k+1} = +1$, monotonicity forces positive labels at every $x \in S^-$ and thus, $S^- \cap \texttt{Fix}^-(Q') = \varnothing$. If $r_{k+1} = -1$, monotonicity forces negative labels at every $y \in S^+$ and thus, $S^+ \cap \texttt{Fix}^+(Q') = \varnothing$. In either case, one of the sets is eliminated.

We now consider the general case of $J \neq \varnothing$. The proof follows a flow similar to that of Lemma 9. We build an induced $\textsc{SafeTarski}^*$ instance on a slice s with $\min\{|J|, |H|\} - 1$ free coordinates. We show that after this induced $\textsc{SafeTarski}^*$ instance is solved, the extended query-answer set Q' must have either eliminated S^- from $\texttt{Fix}^-(Q)$ or S^+ from $\texttt{Fix}^+(Q)$. One subtlety to point out is that $\min\{|J|, |H|\} - 1$ could be 0. In this case it can be checked that the only thing the algorithm needs to do is to query t; after that the updated Q' can achieve the goal.

We start with the case when $|J| \leq |H|$.

First case: $0 < |J| \leq |H|$. We choose any $j \in J$ and define the slice $s \in ([n] \cup \{*\})^k$ by

$$s_\ell = \begin{cases} * & \ell \in J \setminus \{j\} \\ t_\ell & \ell \in H \cup \{j\} \end{cases}.$$

Given any safe function $f \Rightarrow Q$, we use f to induce a function $f' : \mathcal{L}_s \to \mathcal{L}_s \times \{\pm 1\}$ as follows: For any $w \in \mathcal{L}_s$ and any $i \in J \setminus \{j\}$, set $f'(w)_i = f(w)_i$; set $f'(w)_i = w_i$ for every $i \in H \cup \{j\}$; and for coordinate $k+1$, set the label $f'(w)_{k+1}$ by

$$f'(w)_{k+1} = +1 \quad \iff \quad f(w)_j > w_j = t_j \quad \text{or} \quad \big[f(w)_j = t_j \text{ and } f(w)_{k+1} = +1\big].$$

We view f' as a function from $[n]^{|J|-1}$ to $[n]^{|J|-1} \times \{\pm 1\}$ (by renumbering $J \setminus \{j\}$ to $[|J|-1]$ and $k+1$ to $|J|$). It is easy to verify that f' is safe given f is safe. So f' is a valid $\textsc{SafeTarski}^*(n, |J| - 1)$ instance. Moreover, oracle access to f' can be simulated by oracle access to f query by query.

Given Q and oracle access to any safe function $f \Rightarrow Q$, our algorithm proceeds as follows: We run an algorithm $\textsc{Alg}$ for $\textsc{SafeTarski}^*(n, |J| - 1)$ on the unknown safe function f' induced by f. Each time when $\textsc{Alg}$ makes a query $z \in [n]^{|J|-1}$ about f', our algorithm makes a query about f on a point y that corresponds to z as in the construction of f' from f. After receiving $r = f(y)$ from the oracle, our algorithm updates Q by adding a new pair (y, r), replies to $\textsc{Alg}$'s query (based on the construction of f' from f), and continues. This process stops within $\textsc{SafeTarski}^*(n, |J| - 1)$

rounds; at the end, ALG returns a point $z \in [n]^{|J|-1}$ as a solution to f'. Let y be its corresponding point in $[n]^k$. Our algorithm makes one additional query on y (if it has not been queried before). This makes sure that at the end, letting $Q' \supseteq Q$ be the updated query-answer set, Q' must contain a pair (y, r) such that $y \in \mathcal{L}_s$ and one of the following two conditions (with four subcases) holds:

1. $r_i \geq y_i$ for every $i \in J \setminus \{j\}$ and either (1.1) $r_j > t_j$ or (1.2) $r_j = t_j$ and $r_{k+1} = +1$; or
2. $r_i \leq y_i$ for every $i \in J \setminus \{j\}$ and either (2.1) $r_j < t_j$ or (2.2) $r_j = t_j$ and $r_{k+1} = -1$.

(We note that for the special case when $J = \{j\}$, the above can be achieved simply by querying t. Setting $y = t$ satisfies one of the two conditions above.) We show below that Cases (1.1) and (1.2) eliminate S^- from $\texttt{Fix}^-(Q')$, while Cases (2.1) and (2.2) eliminate S^+ from $\texttt{Fix}^+(Q')$.

Consider Cases (1.1) and (1.2). We first show that y is postfixed in J for Q'. To see this, take any monotone function $h \Rightarrow Q'$. Since $(y, r) \in Q'$, we have $h(y)_i \geq y_i$ for every $i \in J \setminus \{j\}$, and also $h(y)_j \geq y_j$ (whether it is (1.1) or (1.2)). So y is postfixed in J for Q'. Assume for a contradiction that $x \in S^- \cap \texttt{Fix}^-(Q')$. Because $x_H \succeq t_H = y_H$, Lemma 11 with $I = J$ implies $y \preceq x$. But given that $x_j \leq t_j = y_j$, we have $x_j = y_j$. We consider the two subcases separately below:

1. **Case 1.1:** $r_j > t_j$. Monotonicity forces $h(x)_j > x_j$ for every monotone function $h \Rightarrow Q'$, contradicting with $x \in \texttt{Fix}(Q')$ by Lemma 6.
2. **Case 1.2:** $r_j = t_j$ and $r_{k+1} = +1$. Monotonicity forces a positive label at x, contradicting with $x \in \texttt{Fix}^-(Q')$ by definition.

So we conclude that $S^- \cap \texttt{Fix}^-(Q') = \varnothing$. Cases (2.1) and (2.2) can be handled similarly.

Second case: $|H| < |J|$. We define the slice $s \in ([n] \cup \{*\})^k$ by

$$s_\ell = \begin{cases} * & \ell \in H \setminus \{k\} \\ t_\ell & \ell \in J \cup \{k\} \end{cases}.$$

Given any safe function $f \Rightarrow Q$, we use f to induce a function $f' : \mathcal{L}_s \to \mathcal{L}_s \times \{\pm 1\}$ as follows: For any $w \in \mathcal{L}_s$ and $i \in H \setminus \{k\}$, set $f'(w)_i = f(w)_i$; set $f'(w)_i = w_i = t_i$ for every $i \in J \cup \{k\}$; and for coordinate $k+1$, set the label $f'(w)_{k+1}$ by

$$f'(w)_{k+1} = +1 \quad \iff \quad f(w)_k > w_k = t_k.$$

We view f' as a function from $[n]^{|H|-1}$ to $[n]^{|H|-1} \times \{\pm 1\}$ (by renumbering $H \setminus \{k\}$ to $[|H|-1]$ and $k+1$ to $|H|$). It is easy to verify that f' is safe given f is safe, so f' is a valid SAFETARSKI$^*(n, |H|-1)$ instance. Similarly, oracle access to f' can be simulated by oracle access to f, query by query.

Our algorithm runs in a similar fashion by simulating an algorithm on SAFETARSKI$^*(n, |H|-1)$. Within SAFETARSKI$^*(n, |H|-1)+1$ many queries, the updated query-answer set Q' must contain a pair (y, r) such that $y \in \mathcal{L}_s$ and one of the following two conditions holds:

1. $r_i \geq y_i$ for every $i \in H \setminus \{k\}$ and $r_k > t_k$; or
2. $r_i \leq y_i$ for every $i \in H$.

Again note that for the special case when $H = \{k\}$, the above can be achieved easily by querying t and setting $y = t$. We show below that Case 1 eliminates S^+, while Case 2 eliminates S^-.

In Case 1, y is postfixed in H for Q'. Assume for a contradiction that $x \in S^+ \cap \texttt{Fix}^+(Q')$. Since $x_J \succeq t_J = y_J$, Lemma 11 with $I = H$ implies $y \preceq x$. But we also have $x_k \leq t_k = y_k$, so $x_k = y_k$.

Monotonicity then forces $h(x)_k > y_k = t_k$ for every monotone function $h \Rightarrow Q'$, contradicting with $x \in \texttt{Fix}(Q')$ by Lemma 6. As a result, we have $S^+ \cap \texttt{Fix}^+(Q') = \varnothing$.

In Case 2, y is prefixed in H for Q'. But every $x \in S^-$ has $x_J \preceq t_J = y_J$ and $x_k \geq t_k + 1 > y_k$. Thus Lemma 11 with $I = H$ excludes every $x \in S^-$ from $\texttt{Fix}(Q')$, giving $S^- \cap \texttt{Fix}^-(Q') = \varnothing$. □

2.8 Putting everything together

Now we are ready to prove Theorem 5.

Proof of Theorem 5. We let Q denote the safe query-answer set maintained by our algorithm. We initialize $Q = \varnothing$ and proceed in rounds while $\mathcal{R}(Q) \neq \varnothing$; the algorithm moves to the next phase once $\mathcal{R}(Q)$ becomes empty. To keep reducing the size of $\mathcal{R}(Q)$, we apply Lemma 10 to obtain a heavy antipodal set-pair (S^-, S^+). Applying Lemma 12 (and using that $k \geq 2$), we can spend at most $\textsc{SafeTarski}^*(n, \lfloor k/2 \rfloor - 1) + 1$ queries to reduce the size of $\mathcal{R}(Q)$ by at least a $1/2^k$ fraction. Given that we trivially have $|\mathcal{R}(Q)| \leq n^{2k}$ initially and queries can only shrink $\mathcal{R}(Q)$, repeating this for at most $\lfloor 2k2^k \log n \rfloor$ rounds suffices to eliminate all bad point-pairs.

Once $\mathcal{R}(Q)$ becomes empty, if either $\texttt{Fix}^-(Q)$ or $\texttt{Fix}^+(Q)$ is empty, we are already done using Lemma 8. Otherwise, the sets are nonempty and coordinate-separable, so we can apply Lemma 9 to solve the rest of the problem with no more than $\textsc{SafeTarski}^*(n, k-1) + 1$ queries.

Hence, for $k \geq 2$ we get the following recurrence:

$$\textsc{SafeTarski}^*(n,k) \leq \textsc{SafeTarski}^*(n,k-1) + 1 + 2k2^k \log n \Big(\textsc{SafeTarski}^*(n, \lfloor k/2 \rfloor - 1) + 1 \Big).$$

Expanding the first term through dimensions $k-1, \ldots, 2$ and using the fact that $\textsc{SafeTarski}^*(n,d)$ is nondecreasing in d, gives a total query cost of at most

$$\textsc{SafeTarski}^*(n,1) + (k-1) + 2\log n \sum_{d=2}^{k} d2^d \Big(\textsc{SafeTarski}^*(n, \lfloor k/2 \rfloor - 1) + 1 \Big).$$

Since $\textsc{SafeTarski}^*(n,1) = O(\log n)$ and $\sum_{d=2}^{k} d2^d = (k-1)2^{k+1}$, we have

$$\textsc{SafeTarski}^*(n,k) \leq O\left(k2^k \log n\right) \cdot \textsc{SafeTarski}^*(n, \lfloor k/2 \rfloor - 1). \tag{3}$$

This finishes the proof of the theorem. □

3 The Lower Bound

In this section, we will prove the query lower bound.

Theorem 3. *For $k \geq 2$, any randomized algorithm for $\textsc{Tarski}(n,k)$ (with probability 1) requires*

$$\Omega\left(\left(\frac{\log n}{\mathsf{polylog}(k)} \right)^{\lfloor \log_4(6k+4) \rfloor} \right)$$

queries in expectation in the worst case, even when the input function has a unique fixed point.

We consider a distribution $\mathcal{F}$ of functions mapping from a hyperrectangle (informally referred to as *cube* going forward) to itself. We will use $Q(\mathcal{F})$ to denote the minimum average-case depth of a deterministic decision tree that always finds a fixed point for a function drawn from the distribution $\mathcal{F}$. By Yao's minimax principle, this gives the same lower bound $Q(\mathcal{F})$ against any randomized algorithm in its worst case.

Our construction is inductive. We use ℓ to denote the level of the induction, which gives us an instance with a query lower bound $\Omega_\ell((\log n)^{\ell+1})$. Throughout the induction, we consider a "base side-length" parameter m, which we will describe later. In the inductive step that lifts from level $\ell-1$ to level ℓ, we compose an outer instance, which is a 1-dimensional Tarski instance with side-length m, and an inner instance, which is a level-$(\ell-1)$ instance. The outer instance is determined by a hidden number t^* in $[m]$, and is defined as $f(t) = t + \mathsf{sign}(t^* - t)$. It is equivalent to a binary search instance for finding a hidden number in $[m]$, where queries provide comparisons between any number and the hidden number, and thus has a query lower bound of $\Omega(\log m)$. This observation allows us to prove a stronger "high-probability" statement for our inductive hypothesis: any deterministic decision tree with depth $q_{\ell,m} = \Theta_\ell((\log m)^{\ell+1})$ can only find a fixed point of a function mapping from $[m^\ell]^{\Theta(4^\ell)}$ to itself with a small probability.

Theorem 6. *There exists an absolute constant $c_0 \in (0, 1/3)$ such that the following holds. Let $m \geq 200$ (base side-length) and $\ell \geq 1$ (level) be integers, and let $n_{\ell,m} = m^\ell$ (side length) and $k_\ell = 2(4^\ell - 1)/3$ (dimension). Then there exists a distribution $\mathcal{F}_{\ell,m}$ of monotone functions $F : [n_{\ell,m}]^{k_\ell} \to [n_{\ell,m}]^{k_\ell}$ such that:*

- *Each function $F \in \mathsf{supp}(\mathcal{F}_{\ell,m})$ has a unique fixed point.*
- *Let $F \sim \mathcal{F}_{\ell,m}$. For any deterministic decision tree with oracle access to F and depth at most $q_{\ell,m} = c_0\left(\frac{\log m}{512\ell^3}\right)^{\ell+1}$, the probability that it finds the fixed point of F is at most $\varepsilon_{\ell,m} = 3m^{-c_0-\frac{1}{8\ell}}$.*

Note that Theorem 6 implies Theorem 3: We let $\ell = \lfloor \log_4(6k+4) \rfloor - 1$. Then, $6k_\ell + 4 = 4^{\ell+1} \leq 6k + 4$, and thus $k_\ell \leq k$. For all sufficiently large n, let $m = \lfloor n^{1/\ell} \rfloor$. Padding the side length from m^ℓ to n and adding $k - k_\ell$ coordinates that are constantly equal to 1 together preserve the monotonicity and the uniqueness of the fixed point. Each query to the padded function can be simulated by one query to the original function. Since $\log m \geq (\log n)/(2\ell)$ and $\varepsilon_{\ell,m} = o(1)$, Theorem 6 gives the following average-case lower bound for any deterministic algorithm.

$$q_{\ell,m}(1 - \varepsilon_{\ell,m}) \geq c_0 \left(\frac{\log n}{1024\ell^4}\right)^{\ell+1} \cdot 0.5 = \Omega\left(\left(\frac{\log n}{\mathsf{polylog}(k)}\right)^{\lfloor \log_4(6k+4) \rfloor}\right).$$

Finally, applying Yao's minimax principle as discussed above, we have proved Theorem 3.

When $\ell = 1$ (the base case), we use the existing construction of [EPRY20] that works for 2-dimensional Tarski. For $\ell \geq 2$, we use an amplification step that lifts the level-$(\ell-1)$ distribution to the level-ℓ distribution, and increases the lower bound by an $\Omega_\ell(\log m)$ factor. We present the step in Section 3.2 and analyze it in Sections 3.3 and 3.4. In the rest of this section, we use $\Delta(A)$ as the set of all distributions supported on A, and identify $\mathbf{1}$ as the all-ones vector.

3.1 The base case: a distribution of two-dimensional monotone functions

Our construction starts from the two-dimensional herringbone construction of Etessami, Papadimitriou, Rubinstein, and Yannakakis [EPRY20]. The construction gives our desired bound for $\ell = 1$ with $k_1 = 2$ dimensions.[5]

Lemma 13 ([EPRY20])**.** *There is a universal constant $c_0 > 0$ such that, for every integer $m \geq 200$, there is a distribution $\mathcal{F}$ of monotone functions $[m]^2 \to [m]^2$, such that*

- *Each function $F \in \mathsf{supp}(\mathcal{F})$ has a unique fixed point.*
- *Let $F \sim \mathcal{F}$. For any deterministic decision tree with oracle access to F and depth at most $q = c_0(\log m)^2$, the probability that it finds the fixed point of F is at most $m^{-c_0 - \frac{1}{8}}$.*

3.2 The amplification step

In this subsection, we present the amplification step of our construction. We fix m as "base side-length" parameter. We will start with a distribution $\mathcal{H}$ whose support consists of monotone functions $H : [n]^k \to [n]^k$ with a unique fixed point. The amplification step will produce a distribution

$$\mathsf{Amp}_m(\mathcal{H}) \in \Delta\left(\left\{f : [m]^2 \times [mn]^{4k} \longrightarrow [m]^2 \times [mn]^{4k}\right\}\right) \tag{4}$$

whose support consists of monotone functions with unique fixed points.

In this subsection, we will use the notation of complement $\overline{z}$ for any $z \in [n]^k$:

$$\overline{z} = (n+1)\mathbf{1} - z.$$

It is clear that $\overline{(\overline{z})} = z$ for any $z \in [n]^k$.

3.2.1 Step 1: putting all candidate fixed points in an anti-chain

Let $H : [n]^k \to [n]^k$ be a function in the support of $\mathcal{H}$. We first define the following function $G_H : [n]^{2k} \to [n]^{2k}$:

$$G_H(x,u) = \left(H(x), \overline{H(\overline{u})}\right), \qquad \forall\ (x,u) \in [n]^{2k}. \tag{5}$$

Suppose that P_H is the unique fixed point of H. A key feature of the function G_H is that it has a unique fixed point, as shown by the following lemma.

Lemma 14. *For any $x, u \in [n]^k$, we have $G_H(x,u) = (x,u)$ if and only if $x = P_H$ and $u = \overline{P_H}$.*

Proof. If $G_H(x,u) = (x,u)$, we have $H(x) = x$ and $\overline{H(\overline{u})} = u$. The second equality gives us $H(\overline{u}) = \overline{u}$. Since P_H is the unique fixed point of H, we have $x = P_H$ and $\overline{u} = P_H$, and thus $u = \overline{P_H}$.

Conversely, if we have $x = P_H$ and $u = \overline{P_H}$, we have

$$G_H(x,u) = (H(P_H), \overline{H(P_H)}) = (P_H, \overline{P_H}) = (x,u)\ .$$

Hence, we have completed the proof. □

[5] Although [EPRY20] only explicitly states an $\Omega((\log m)^2)$ randomized query lower bound, the stronger probability bound in Lemma 13 also follows from their construction and analysis.

Lemma 14 implies that the unique fixed point must lie in

$$C = \{(z, \overline{z}) : z \in [n]^k\} \subseteq \left\{ x \in [n]^{2k} : \sum_{i \in [2k]} x_i = k(n+1) \right\} .$$

Because $\{x \in [n]^{2k} : \sum_{i\in[2k]} x_i = k(n+1)\}$ is an anti-chain, after this step, we can ensure that all potential fixed points of G_H are in the fixed anti-chain C (that is independent of the function H).

3.2.2 Step 2: Hiding one new bit in a sign function

We partition the space $[n]^{2k}$ into two subsets: D_C and U_C, where the set D_C consists of all points that are below some point in the anti-chain C and U_C consists of all other points.

$$D_C = \{w \in [n]^{2k} : \exists w' \in C, w \preceq w'\}, \quad \text{and} \quad U_C = [n]^{2k} \setminus D_C.$$

We remark here that $C \subseteq D_C$ and that both D_C and U_C are independent of H.

Next, fixing a function $H \in \mathsf{supp}(\mathcal{H})$, we consider hiding one bit $c \in \{0, 1\}$ at the unique fixed point of G_H. More specifically, we consider the following sign function $\mathsf{S}_{H,c} : [n]^{2k} \to \{0, 1\}$:

$$\forall X \in [n]^{2k}, \quad \mathsf{S}_{H,c}(X) = \begin{cases} c, & X = (P_H, \overline{P_H}), \\ 0, & X \in D_C \setminus \{(P_H, \overline{P_H})\}, \\ 1, & X \in U_C. \end{cases} \tag{6}$$

Intuitively, this sign function is hiding the bit c unless you have reached the fixed point of H: for all other points, the function value is automatically 0 if it is in D_C, while the value is 1 if it is in U_C. Therefore, to learn c from querying $\mathsf{S}_{H,c}$, the only way is to find the fixed point P_H of H and query $X = (P_H, \overline{P_H})$.

Another key property of this sign function is that it is trivially monotone because D_C includes all points that are below some point of D_C. We will formalize and use this property later in proving that our constructed distribution of functions is monotone (Lemma 16).

3.2.3 A failed attempt to put the first two steps together

With the first two steps, one may be curious if we can put them together in a $2k + 1$-dimensional instance, with the outer 1-dimensional Tarski instance encoded in the first coordinate and the inner instances G_H embedded in the last $2k$ coordinates.

More specifically, we consider constructing a function $F : [m] \times [mn]^{2k} \to [m] \times [mn]^{2k}$. We call the first coordinate $a \in [m]$ the "sign coordinate" and the last $2k$ coordinates $X \in [mn]^{2k}$ the "value coordinates". Accordingly, the function value is partitioned into $F(\cdot) = (F_a(\cdot), F_X(\cdot))$ with $F_a(\cdot) \in [m]$ and $F_X(\cdot) \in [mn]^{2k}$.

We consider m boxes for the "value coordinates". The t-th box is defined as all points whose coordinates are between $(t-1) \cdot n + 1$ and $t \cdot n$, i.e.,

$$B_t = \{X \in [mn]^{2k} : \forall i \in [2k], (t-1) \cdot n < X_i \leq t \cdot n\}, \tag{7}$$

which forms a $2k$-dimensional hypercube of side-length n and it is straightforward to consider a translated copy of G_H (for $H \in \mathsf{supp}(\mathcal{H})$) as a function mapping from any B_t to itself. For the value coordinates, we sample $H_1, \ldots, H_m \sim \mathcal{H}$ from the original distribution $\mathcal{H}$, and define the

value $F_X(a, X)$ as the value of $\mathcal{P}_a(X)$ in G_{H_a}'s translated copy, where $\mathcal{P}_a(X)$ is the projection of X onto box B_a. This is a natural way to guarantee the monotonicity for the value coordinates (because any point $W \in B_a$ is below any point $W' \in B_{a+1}$).

At the same time, we consider hiding a random number $t^* \sim \mathsf{uniform}([m])$ in the outer 1-dimensional instance. We will use this outer instance to define the displacement $F_a(a, X) - a$ for each point. Ideally, we want to place the sign $\mathsf{sign}(t^* - a)$ on the "fixed point" of each box (points (a, X) with $X = FP_a$, where FP_a is the fixed point for the translated copy of G_{H_a} in B_a), and place default values independent of t^* at all other points, similarly to Equation (6). However, this requires three possible displacement values $-1, 0, 1$ instead of the two values $0, 1$ used in Equation (6). Next, we argue that this could be problematic to guarantee the monotonicity of the sign coordinate.

Let MIN_a and MAX_a be the minimum and maximum elements of B_a, respectively. Because $\mathsf{MIN}_a \preceq FP_a \preceq \mathsf{MAX}_a$, the monotonicity of F requires our definition of F_a to satisfy $F_a(a, \mathsf{MIN}_a) \leq F_a(a, FP_a) \leq F_a(a, \mathsf{MAX}_a)$. For $1 < a < m$, it is possible that the displacement $F_a(a, FP_a) - a$ is $-1, 0$ or 1, depending on our choice of t^*. Therefore, the default value of $F_a(a, \mathsf{MIN}_a)$ must be at most $a - 1$, while the default value of $F_a(a, \mathsf{MAX}_a)$ must be at least $a + 1$. However, this could lead to a violation of monotonicity in other pairs of points. Consider two points for some $t \in [2, m - 2]$:

$$W = (t, \mathsf{MAX}_t) \quad \text{and} \quad W' = (t + 1, \mathsf{MIN}_{t+1}) \ .$$

We have $W \preceq W'$. According to the above discussion, because these default values must reveal no information about t^*, we have $F_a(W) \geq t + 1$ and $F_a(W') \leq (t + 1) - 1 = t$, which violate $F(W) \preceq F(W')$.

3.2.4 Step 3: Defining the new distribution of monotone functions

In this step, we define the new distribution of monotone functions. Beyond our failed attempt discussed earlier, we consider $4k + 2$-dimensional instances, which can be regarded as doubling the failed construction. The reason we can circumvent the monotonicity issue in the failed attempt is that we can use displacement values in $\{0, 1\}$ on one dimension and use $\{-1, 0\}$ on another. Because we don't have displacement $-1, 1$ on the same dimension, this helps to rule out the violation we discussed earlier. To prevent introducing multiple fixed points, we then swap the updated values of the two sign coordinates.

In this construction, we will call the first two coordinates "sign coordinates" and the last $4k$ coordinates "value coordinates". We divide the "value coordinate" into two groups: the first consists of the first $2k$ value coordinates, and the second consists of the last $2k$ value coordinates. In addition, we consider m boxes that are restricted to each group. We use the same definition of B_t as in Equation (7). Formally, we define $\mathcal{P}_t(X)$ as the projection of $X \in [mn]^{2k}$ onto the box B_t as follows:

$$[\mathcal{P}_t(X)]_i = \min\{t \cdot n, \max\{(t - 1) \cdot n + 1, X_i\}\} \ .$$

Next, we construct a random function $F : [m]^2 \times [mn]^{4k} \to [m]^2 \times [mn]^{4k}$. The distribution of the function is then our desired $\mathsf{Amp}_m(\mathcal{H})$.

1. We first sample m i.i.d. functions $H_1, H_2, \ldots, H_m$ from the original distribution $\mathcal{H}$. For simplicity, we write P_i for the unique fixed point of H_i.

2. We sample a hidden number t^* from $[m]$ uniformly at random. We define the sign $c_t = \mathbf{1}(t \leq t^*)$ for each $t \in [m]$. We also define $c_{m+1} = 0$ for simplicity.

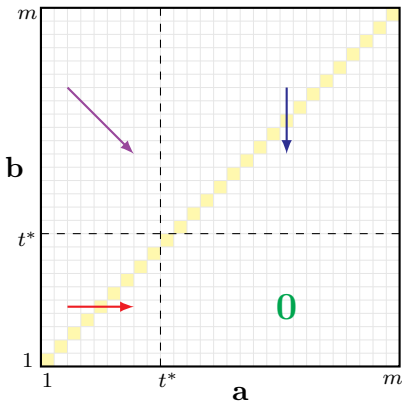


Figure 1: Illustration of updates on the sign coordinates a, b before swapping them, when the translated projections of X and Y are fixed points of G_{H_a} and G_{H_b}. We increase a by one when $a < t^*$ and decrease b by one when $b > t^*$. Hence, the bottom-left region moves right, the top-right region moves down, the top-left region moves both right and down, and the bottom-right region remains unchanged. The yellow cells mark the diagonal $a = b$, which is fixed under our swapping operation. This illustrates how the update and swap rules single out (t^*, t^*) as the sign coordinates of the unique fixed point of our constructed functions.

3. For each $W \in [m]^2 \times [mn]^{4k}$, we rewrite $W = (a, b, X, Y)$ where $a, b \in [m]$ and $X, Y \in [mn]^{2k}$.

4. For each $W \in [m]^2 \times [mn]^{4k}$, we rewrite $F(W) = (F_a(W), F_b(W), F_X(W), F_Y(W))$ where $F_a(W), F_b(W) \in [m]$ and $F_X(W), F_Y(W) \in [mn]^{2k}$.

5. **Defining the function value for the "value coordinates":** we pair a with X and b with Y. The value of F_X is determined by the pair (a, X), while the value of F_Y is determined by the pair (b, Y). The formulations for F_X and F_Y are identical based on the pair. For F_X, we first compute the projection of X onto B_a, translate it to the space $[n]^{2k}$, and then consider its value of G_{H_a} (translated back to the space B_a). More specifically, we define

$$F_X(W) = (a-1) \cdot n \cdot \mathbf{1} + G_{H_a}(\mathcal{P}_a(X) - (a-1) \cdot n \cdot \mathbf{1}) \ ,$$
$$F_Y(W) = (b-1) \cdot n \cdot \mathbf{1} + G_{H_b}(\mathcal{P}_b(Y) - (b-1) \cdot n \cdot \mathbf{1}) \ .$$

6. **Defining the function value for the "sign coordinates":** We will hide the sign c_t (comparing t with t^*) in these coordinates. As in the definition for the "value coordinates", we still pair (a, X) and (b, Y). We consider slightly updating the "sign coordinates" (also demonstrated in Figure 1) and then swapping them to define their function value, following the rules below:

   - We move a based on X's sign in the sign function $\mathsf{S}_{H_a, c_{a+1}}$ and place its value on F_b:

   $$F_b(W) = \min\{m, a + \mathsf{S}_{H_a, c_{a+1}}(\mathcal{P}_a(X) - (a-1) \cdot n \cdot \mathbf{1})\}.$$

   Let $X' = \mathcal{P}_a(X) - (a-1)n \cdot \mathbf{1}$ be translated projection of X. This gives $F_b(W) = a$ if $X' \in D_C \setminus \{(P_a, \overline{P_a})\}$ or $F_b(W) = \min\{m, a+1\}$ if $X' \in U_C$. More importantly, when X' equals the fixed point $(P_a, \overline{P_a})$, this gives $F_b(W) = a$ when $a \geq t^*$ but gives $F_b(W) = a + 1$ when $a < t^*$.

- We move b based on Y's sign in the sign function S_{H_b,c_b} and place its value on F_a:

$$F_a(W) = \max\{1, b-1+\mathsf{S}_{H_b,c_b}(\mathcal{P}_b(Y)-(b-1)\cdot n\cdot \mathbf{1})\}.$$

  Let $Y' = \mathcal{P}_b(Y) - (b-1)n\cdot\mathbf{1}$ be the translated projection of Y. This gives $F_a(W) = \max\{1, b-1\}$ if $Y' \in D_C \setminus \{(P_b, \overline{P_b})\}$ or $F_a(W) = b$ if $Y' \in U_C$. More importantly, when Y' equals the fixed point $(P_b, \overline{P_b})$, this gives $F_a(W) = b$ when $b \le t^*$ but gives $F_a(W) = b-1$ when $b > t^*$.

Overall, we conclude the following intuition for the definition on the "sign coordinates":

- We want F_a (or, F_b) to reveal no information about t^* if the translated projection of Y (or, X) is not a fixed point.
- Assume that the translated projections of X, Y are both fixed points. The swapping idea ensures that a, b should be equal to each other in any fixed point W of F. This ensures that the fixed point is unique (and thus finding it requires finding the hidden t^*). Then, we only move exactly one of a and b in one (fixed) direction when $a = b \neq t^*$, in order to guarantee monotonicity.

3.3 Basic properties of the constructed distribution of functions

In this subsection, we show that any function in the support of the constructed distribution $\mathsf{Amp}_m(\mathcal{H})$ is monotone and has a unique fixed point.

3.3.1 Uniqueness of the fixed point

Lemma 15. *Consider any $F \in \mathsf{supp}(\mathsf{Amp}_m(\mathcal{H}))$. Suppose that the sampled m functions from $\mathcal{H}$ are $H_1, H_2, \ldots, H_m$, and their unique fixed points are respectively $P_1, P_2, \cdots, P_m \in [n]^k$. Suppose that t^* is the sampled hidden number in $[m]$. Let*

$$P^* = (P_{t^*} + (t^*-1)n\cdot\mathbf{1}, \overline{P_{t^*}} + (t^*-1)n\cdot\mathbf{1}).$$

Then, F has the unique fixed point (t^*, t^*, P^*, P^*).

Proof. First, we show that $W^* = (t^*, t^*, P^*, P^*)$ is a fixed point by simple calculations. Since $P_{t^*} \in [n]^k$, we have $P^* \in B_{t^*}$ by the definition of B_{t^*} and thus $\mathcal{P}_{t^*}(P^*) = P^*$. Hence, we have $\mathcal{P}_{t^*}(P^*) - (t^*-1)n\cdot\mathbf{1} = (P_{t^*}, \overline{P_{t^*}})$. Because we have $G_{H_{t^*}}(P_{t^*}, \overline{P_{t^*}}) = (P_{t^*}, \overline{P_{t^*}})$, we have $F_X(W^*) = P^*$ and $F_Y(W^*) = P^*$ by their definitions. By definitions of F_a and F_b, we have

$$F_b(W^*) = \min\{m, t^* + c_{t^*+1}\} = t^* \ , \ \textit{and} \qquad (c_{t^*+1} = 0 \text{ as } t^*+1 > t^*)$$
$$F_a(W^*) = \max\{1, t^* - 1 + c_{t^*}\} = t^* \ . \qquad (c_{t^*} = 1 \text{ as } t^* \le t^*)$$

Therefore, we have $F(W^*) = W^*$. Next, we show that this is the only fixed point of F. Let $W = (a, b, X, Y)$ be any fixed point of F.

First, we prove the characterization of X, Y: $X = (P_a, \overline{P_a}) + (a-1)n\cdot\mathbf{1}$ and $Y = (P_b, \overline{P_b}) + (b-1)n\cdot\mathbf{1}$. Since $X = F_X(W) \in B_a$ and $Y = F_Y(W) \in B_b$, we have $X \in B_a$ and $Y \in B_b$ and thus $\mathcal{P}_a(X) = X$ and $\mathcal{P}_b(Y) = Y$. According to the definitions of F_X, F_Y, we have

$$X = F_X(W) = (a-1)n\cdot\mathbf{1} + G_{H_a}\big(X - (a-1)n\cdot\mathbf{1}\big)$$
$$Y = F_Y(W) = (b-1)n\cdot\mathbf{1} + G_{H_b}\big(Y - (b-1)n\cdot\mathbf{1}\big).$$

This implies that $X-(a-1)n\cdot\mathbf{1}$ is a fixed point of G_{H_a} and $Y-(b-1)n\cdot\mathbf{1}$ is a fixed point of G_{H_b}. By Lemma 14, we have obtained the characterization of X, Y.

Second, we show that $a=b=t^*$. Given the characterization of X and Y, we have

$$\mathcal{P}_a(X)-(a-1)n\cdot\mathbf{1}=(P_a,\overline{P_a}),\quad\text{and}\quad\mathcal{P}_b(Y)-(b-1)n\cdot\mathbf{1}=(P_b,\overline{P_b}).$$

According to the definition of F_a, F_b, we have

$$a=F_a(W)=\max\{1,b-1+c_b\},\quad\text{and}\quad b=F_b(W)=\min\{m,a+c_{a+1}\}. \tag{8}$$

Suppose that $a<t^*$. Then, we have $c_{a+1}=1$ and thus $b=a+1\le t^*$. Therefore, $a=\max\{1,b\}=b$, contradicting $b=a+1$. Suppose that $a>t^*$. Then, we have $c_{a+1}=0$ and thus $b=a>t^*$. Therefore, $a=b-1$, contradicting $b=a$. Therefore, we have $a=t^*$. Then, $c_{a+1}=0$ and we get $b=a$.

In conclusion, we have $a=b=t^*$ and $X=Y=(P_{t^*},\overline{P_{t^*}})+(t^*-1)n\cdot\mathbf{1}=P^*$. This gives the uniqueness of the fixed point of F. □

3.3.2 Monotonicity

Lemma 16. *Every member $F\in\mathsf{supp}(\mathsf{Amp}_m(\mathcal{H}))$ is monotone.*

Proof. First, we prove that G_H is monotone for any $H\in\mathsf{supp}(\mathcal{H})$. Suppose that $(x,u)\preceq(x',u')$. Because of the monotonicity of H, we have $H(x)\preceq H(x')$. Also, $u\preceq u'$ implies that $\bar{u}\succeq\bar{u}'$, and hence $H(\bar{u})\succeq H(\bar{u}')$. Taking complements gives $\overline{H(\bar{u})}\preceq\overline{H(\bar{u}')}$. By definition of G_H, we have $G_H(x,u)\preceq G_H(x',u')$.

Next, we prove that $\mathsf{S}_{H,c}$ is monotone for any $H\in\mathsf{supp}(\mathcal{H})$ and $c\in\{0,1\}$. Let $Z\preceq Z'\in[n]^{2k}$. If $\mathsf{S}_{H,c}(Z)=0$ or $Z=Z'$, it is clear that $\mathsf{S}_{H,c}(Z)\le\mathsf{S}_{H,c}(Z')$. Otherwise, we have either $Z\in U_C$, or $Z=(P_H,\overline{P_H})$ and $c=1$. In the former case, if $Z'\in D_C$, we can find $Z''\in C$ such that $Z'\preceq Z''$ and thus $Z\preceq Z''$. This implies $Z\in D_C$ by the definition of D_C, which contradicts that $Z\in U_C$ and U_C, D_C are disjoint. In the latter case, $Z'\in U_C$ as well; otherwise, $Z\prec Z'\preceq Z''$ for some $Z''\in C$, contradicting that $Z\in C$ and C forms an anti-chain. Thus $\mathsf{S}_{H,c}(Z')=1=\mathsf{S}_{H,c}(Z)$ in either case.

We now use these two properties to prove that F is monotone. It suffices to compare two points $W\preceq W'$ that differ in exactly one coordinate, because of the transitivity of $\preceq$. We write $W=(a,b,X,Y)$ and discuss the following cases.

Only a increases. Write $W'=(a',b,X,Y)$ with $a'>a$. The outputs F_a and F_Y are unchanged. Since every sign function takes values in $\{0,1\}$,

$$F_b(W)\le\min\{m,a+1\}\le a'\le F_b(W').$$

Moreover, because $F_X(W)\in B_a$ and $F_X(W')\in B_{a'}$, for every $i\in[2k]$,

$$[F_X(W)]_i\le an<(a'-1)n+1\le[F_X(W')]_i.$$

Therefore, $F(W)\preceq F(W')$.

Only b increases. Write $W'=(a,b',X,Y)$ with $b'>b$. The outputs F_b and F_X are unchanged, while

$$F_a(W)\le b\le b'-1\le F_a(W').$$

Moreover, because $F_Y(W) \in B_b$ and $F_Y(W') \in B_{b'}$, for every $i \in [2k]$,

$$[F_Y(W)]_i \le bn < (b'-1)n + 1 \le [F_Y(W')]_i.$$

Therefore, $F(W) \preceq F(W')$.

Only one coordinate of X increases. Write $W' = (a, b, X', Y)$, where $X \prec X'$. The outputs F_a and F_Y are unchanged. Since $\mathcal{P}_a$ and G_{H_a} are monotone, we have $F_X(W) \preceq F_X(W')$ by the definition of F_X. Moreover, because of the monotonicity of $\mathsf{S}_{H_a,c_{a+1}}$, we have

$$\begin{aligned} F_b(W) &= \min\{m, a + \mathsf{S}_{H_a,c_{a+1}}(\mathcal{P}_a(X) - (a-1)n \cdot \mathbf{1})\} \\ &\le \min\{m, a + \mathsf{S}_{H_a,c_{a+1}}(\mathcal{P}_a(X') - (a-1)n \cdot \mathbf{1})\} = F_b(W'). \end{aligned}$$

Only one coordinate of Y increases. Write $W' = (a, b, X, Y')$, where $Y \prec Y'$. The outputs F_b and F_X are unchanged. Since P_b and G_{H_b} are monotone, we have $F_Y(W) \preceq F_Y(W')$ by the definition of F_Y. Moreover, because of the monotonicity of S_{H_b,c_b}, we have

$$\begin{aligned} F_a(W) &= \max\{1, b - 1 + \mathsf{S}_{H_b,c_b}(\mathcal{P}_b(Y) - (b-1)n \cdot \mathbf{1})\} \\ &\le \max\{1, b - 1 + \mathsf{S}_{H_b,c_b}(\mathcal{P}_b(Y') - (b-1)n \cdot \mathbf{1})\} = F_a(W'). \end{aligned}$$

Therefore, we have $F(W) \preceq F(W')$ in every case, proving that F is monotone. □

3.4 Query lower bound for the constructed distribution of functions

In this subsection, we analyze the amplification step and prove Theorem 6, restated below.

Theorem 6. *There exists an absolute constant $c_0 \in (0, 1/3)$ such that the following holds. Let $m \ge 200$ (base side-length) and $\ell \ge 1$ (level) be integers, and let $n_{\ell,m} = m^\ell$ (side length) and $k_\ell = 2(4^\ell - 1)/3$ (dimension). Then there exists a distribution $\mathcal{F}_{\ell,m}$ of monotone functions $F : [n_{\ell,m}]^{k_\ell} \to [n_{\ell,m}]^{k_\ell}$ such that:*

- *Each function $F \in \mathsf{supp}(\mathcal{F}_{\ell,m})$ has a unique fixed point.*
- *Let $F \sim \mathcal{F}_{\ell,m}$. For any deterministic decision tree with oracle access to F and depth at most $q_{\ell,m} = c_0\big(\frac{\log m}{512\ell^3}\big)^{\ell+1}$, the probability that it finds the fixed point of F is at most $\varepsilon_{\ell,m} = 3m^{-c_0 - \frac{1}{8\ell}}$.*

Suppose that $\mathcal{H}$ is the level-$(\ell-1)$ distribution and satisfies the level-$(\ell-1)$ guarantee in Theorem 6. We will show that a padded version of our amplification $\mathcal{F} = \mathsf{Amp}_m(\mathcal{H})$ satisfies the level-ℓ guarantee. To make the notation simpler (hiding the subscript), we say functions in $\mathsf{supp}(\mathcal{H})$ map from $[n]^k$ to itself, and the induction hypothesis is

> **Induction hypothesis:** *the probability that any deterministic decision tree with depth at most q finds the fixed point of $H \sim \mathcal{H}$ is at most ε.*

Recall our notations in Section 3.2. $H_1, \cdots, H_m$ are i.i.d. samples from $\mathcal{H}$ and $t^* \sim \mathsf{uniform}([m])$. $P_1, \cdots, P_m$ are the unique fixed points of $H_1, \cdots, H_m$ respectively. We consider the signs $c_t = \mathbf{1}(t \le t^*)$ and $c_{m+1} = 0$. Then, we summarize the definition $F(W) = (F_a(W), F_b(W), F_X(W), F_Y(W))$ for $W = (a, b, X, Y)$ in terms of $H_1, \ldots, H_m$ and $c_1, \ldots, c_{m+1}$ as follows.

- $F_X(W)$ uses a, X. Let $\mathcal{P}_a(X) - (a-1)n \cdot \mathbf{1} = (x, u)$. We make two queries $H_a(x)$ and $H_a(\overline{u})$. The values are then determined by these queries.
- $F_Y(W)$ uses b, Y. Let $\mathcal{P}_b(Y) - (b-1)n \cdot \mathbf{1} = (y, v)$. We make two queries $H_b(y)$ and $H_b(\overline{v})$. The values are then determined by these queries.
- $F_a(W)$ uses b, Y. The value is then determined by c_b only when $\mathcal{P}_b(Y) - (b-1)n \cdot \mathbf{1}$ is the unique fixed point of G_{H_b} (whether this holds can be determined by the values of $F_Y(W)$). Otherwise, it has a default value (based on Y).
- $F_b(W)$ uses a, X. The value is then determined by c_{a+1} only when $\mathcal{P}_a(X) - (a-1)n \cdot \mathbf{1}$ is the unique fixed point of G_{H_a} (whether this holds can be determined by the values of $F_X(W)$). Otherwise, it has a default value (based on X).

Next, we present the key lemma in our analysis that establishes a similar probability bound for $\mathsf{Amp}_m(\mathcal{H})$.

Lemma 17. *Assume the induction hypothesis for $\mathcal{H}$. For every deterministic decision tree of depth at most $q' = \frac{q \log m}{64}$, the probability that it finds the fixed point of $F \sim \mathsf{Amp}_m(\mathcal{H})$ is at most $\varepsilon' = \frac{1}{\sqrt{m}} + \frac{q\varepsilon}{3}$.*

Proof. Consider any deterministic decision tree of depth at most q'. For a query $W = (a, b, X, Y)$, using our previous discussions, we can make at most 4 queries in total to $H_1, \ldots, H_m$ to determine $F_X(W)$ and $F_Y(W)$. Further, the value of $F_b(W)$ (or $F_a(W)$) is independent of any sign c_t unless X's projection $\mathcal{P}_a(X) - (a-1)n \cdot \mathbf{1}$ (resp., for $\mathcal{P}_b(Y) - (b-1)n \cdot \mathbf{1}$) equals $(P_a, \overline{P_a})$ (resp., for $(P_b, \overline{P_b})$), where P_a (resp., P_b) is the unique fixed point of H_a (resp., H_b). This means that a query to $F(W)$ only reveals at most four queries in total, to $H_1, \cdots, H_m$; and the number of different signs c_t revealed by $F(W)$ is upper bounded by the number of queries with different (a, X) such that $\mathcal{P}_a(X) - (a-1)n \cdot \mathbf{1} = (P_a, \overline{P_a})$ plus the number of queries with different (b, Y) such that $\mathcal{P}_b(Y) - (b-1)n \cdot \mathbf{1} = (P_b, \overline{P_b})$.

On the other hand, if the decision tree successfully finds the unique fixed point of F, it must learn t^* because of Lemma 15. Therefore, we can upper bound the probability that the decision tree finds the fixed point of F by the probability that it correctly finds t^*.

Let R denote the random variable representing the total number of different signs c_t revealed by the queries to F.

First, we upper bound the probability of learning t^* if the queries reveal at most $0.5 \log m$ signs $c_t \in \{0, 1\}$. We observe that because $H_1, \cdots, H_m$ are sampled independently of t^*, the values of the revealed queries to them do not reveal any information about t^*. Therefore, conditioning on $H_1, \ldots, H_m$, because any deterministic decision tree with at most d 0-1 queries can have at most 2^d different outputs, we have

$$\Pr\left[\begin{array}{c}\text{the decision tree finds } t^* \\ \text{and } R \le 0.5 \log m\end{array} \middle| H_1, \ldots, H_m\right] \le \frac{1}{m} \cdot 2^{0.5 \log m} = \frac{1}{\sqrt{m}}.$$

Taking the average over all possibilities of $H_1, \cdots, H_m$, we can show that the probability that the decision tree finds t^* and $R \le 0.5 \log m$ is at most $1/\sqrt{m}$.

Second, we upper bound the probability that the decision tree with depth q' reveals $> 0.5 \log m$ signs, i.e., $\Pr[R > 0.5 \log m]$. Recall that we can obtain each $F(W)$ by making 4 queries, each to one of $H_1, \ldots, H_m$. Hence, we can transform each query to $F(W)$ into 4 such queries. Let

$\mathcal{Q}$ denote the ordered list of transformed queries. For each $t \in [m]$, we let E_t be the event that we query P_t (the unique fixed point of H_t) to H_t in $\mathcal{Q}$ and we let N_t denote the total number of queries to H_t in $\mathcal{Q}$. Note that the sign c_t is revealed only when we query with $(a, \mathcal{P}_a(X)) = \big(t-1,\ (P_{t-1}, \overline{P_{t-1}}) + (t-2)n \cdot \mathbf{1}\big)$ (if $t > 1$) or $(b, \mathcal{P}_b(Y)) = \big(t,\ (P_t, \overline{P}_t) + (t-1)n \cdot \mathbf{1}\big)$. In these cases, we query P_{t-1} to H_{t-1} or query P_t to H_t in $\mathcal{Q}$. When c_t is revealed, either E_{t-1} (if $t > 1$) or E_t occurs. Therefore, we have $R \le 2\sum_{t\in[m]} \mathbf{1}(E_t)$ and thus

$$\Pr[R > 0.5 \log m] \le \Pr\left[\sum_{t\in[m]} \mathbf{1}(E_t) > \frac{1}{4}\log m\right].$$

Because the depth of the decision tree is at most q', the number of queries in $\mathcal{Q}$ is at most $4q'$, which means that $\sum_{t\in[m]} N_t \le 4q'$. The number of $t \in [m]$ such that $N_t \ge q$ is at most $4q'/q$. Also, note that when $N_t = 0$, E_t can't happen. For t such that $N_t \ge 1$, we let E'_t denote the event that there is a query asking for $H_t(P_t)$ among the first $\le q-1$ queries to H_t. Hence, we have

$$\begin{aligned}\Pr\left[\sum_{t\in[m]} \mathbf{1}(E_t) > \frac{1}{4}\log m\right] &\le \Pr\left[\sum_{t\in[m]:1\le N_t<q} \mathbf{1}(E_t) > \frac{1}{4}\log m - \frac{4q'}{q}\right] \\ &\le \Pr\left[\sum_{t\in[m]:N_t\ge 1} \mathbf{1}(E'_t) > \frac{1}{4}\log m - \frac{4q'}{q}\right].\end{aligned}$$

Consider any $t \in [m]$ such that $N_t \ge 1$. Because $H_1, \ldots, H_m$ (except H_t) and t^* are sampled independently of H_t, we can simulate the following (randomized) decision tree that makes at most $q-1$ queries to H_t. Every time a transformed query is about H_t, we query it until we have asked $q-1$ queries in total; otherwise, we consider it as some random bits, taken over the randomness of $H_1, \ldots, H_m$ (except H_t) and t^*; finally, we check if we have queried the fixed point of H_t by the queries: if we have asked a point x such that $H_t(x) = x$, we output the point; otherwise, we output arbitrarily. According to our induction hypothesis, any decision tree with depth at most q can only find the fixed point of H_t with probability at most ε. Note that this simulated decision tree finds the fixed point when E'_t happens. Therefore, we have

$$\mathbb{E}\left[\sum_{t\in[m]:N_t\ge 1} \mathbf{1}(E'_t)\right] = \sum_{t\in[m]} \Pr[E'_t \text{ and } N_t \ge 1] \le \varepsilon \sum_{t\in[m]} \mathbb{E}[N_t] \le 4q'\varepsilon\ .$$

Using Markov's inequality, we have

$$\begin{aligned}\Pr\left[\sum_{t\in[m]:N_t\ge 1} \mathbf{1}(E'_t) > \frac{1}{4}\log m - \frac{4q'}{q}\right] &\le \frac{4q'\varepsilon}{\frac{1}{4}\log m - \frac{4q'}{q}} \\ &= \frac{(q\varepsilon \log m)/16}{\frac{1}{4}\log m - \frac{1}{16}\log m} = \frac{q\varepsilon}{3}\ .\end{aligned}$$

Hence, we have shown that $\Pr[R > 0.5 \log m] \le \frac{q\varepsilon}{3}$.

Finally, putting everything together, we have the following upper bound for the probability that the decision tree finds t^*:

$$\Pr\left[\text{the decision tree finds } t^*\right] \le$$
$$\Pr[R > 0.5 \log m] + \Pr\left[\text{the decision tree finds } t^* \text{ and } R \le 0.5 \log m\right] \le \frac{1}{\sqrt{m}} + \frac{q\varepsilon}{3} \ .$$

Because the probability that the decision tree finds the fixed point of F is upper bounded by the above probability, we have completed the proof. □

Proof of Theorem 6. Fix any $m \ge 200$. The base case ($\ell = 1$) follows from Lemma 13.

Let $\ell \ge 2$ and suppose that the theorem holds at level $\ell - 1$. Let $\mathcal{H} = \mathcal{F}_{\ell-1,m}$ and apply the amplification step to obtain $\widehat{\mathcal{F}} = \mathsf{Amp}_m(\mathcal{H})$. Because we have $n_{\ell,m} = m n_{\ell-1,m}$ and $k_\ell = 4k_{\ell-1} + 2$ by their definitions, every $\widehat{F} \in \mathsf{supp}(\widehat{\mathcal{F}})$ maps $[m]^2 \times [n_{\ell,m}]^{k_\ell - 2}$ to itself. By Lemmas 15 and 16, every such $\widehat{F}$ is monotone and has a unique fixed point.

To obtain $\mathcal{F}_{\ell,m}$, we pad the first two coordinates that have side length m. For each $W \in [n_{\ell,m}]^{k_\ell}$, we let $F(W) = \widehat{F}(\Pi(W))$, where we sample $\widehat{F} \sim \widehat{\mathcal{F}}$ and define

$$\Pi(W) = \left(\min\{W_1, m\}, \min\{W_2, m\}, W_3, \ldots, W_{k_\ell}\right)$$

The function F is monotone because both Π and $\widehat{F}$ are monotone. Moreover, F and $\widehat{F}$ have the same unique fixed point because any point $W \notin [m]^2 \times [n_{\ell,m}]^{k_\ell - 2}$ maps to $F(W) \in [m]^2 \times [n_{\ell,m}]^{k_\ell - 2}$.

It remains to prove the probability bound (the second bullet of the theorem). Let $q = q_{\ell-1,m}$ and $\varepsilon = \varepsilon_{\ell-1,m}$. Using the definition of $q_{\ell,m}$, we have

$$\frac{q_{\ell,m}}{(q_{\ell-1,m} \log m)/64} = \frac{1}{8\ell^3} \left(\frac{\ell - 1}{\ell}\right)^{3\ell} \le 1.$$

Due to Lemma 17, any deterministic decision tree of depth at most $q_{\ell,m}$ can find the fixed point of $\widehat{F} \sim \widehat{\mathcal{F}}$ with probability at most

$$\begin{aligned}
\frac{2}{\sqrt{m}} + \frac{q_{\ell-1,m}\varepsilon_{\ell-1,m}}{3} &= \frac{2}{\sqrt{m}} + c_0 \left(\frac{\log m}{512(\ell-1)^3}\right)^{\ell} \cdot m^{-c_0 - \frac{1}{8(\ell-1)}} \\
&\le 2m^{-c_0 - \frac{1}{8\ell}} + c_0 \left(\frac{\log m}{512(\ell-1)^3}\right)^{\ell} m^{-\frac{1}{8\ell(\ell-1)}} \cdot m^{-c_0 - \frac{1}{8\ell}} \ .
\end{aligned}$$

Because we can show that $\left(\frac{\log m}{512(\ell-1)^3}\right)^{\ell} < m^{1/8\ell(\ell-1)}$ by the following calculation:

$$\begin{aligned}
\log\left(\left(\frac{\log m}{512(\ell-1)^3}\right)^{\ell}\right) &\le \ell \cdot \frac{\log m}{512(\ell-1)^3} && (\log x \le x \text{ for } x > 0) \\
&= \frac{\ell}{512(\ell-1)^3} \cdot \log m < \frac{1}{8\ell(\ell-1)} \cdot \log m \ , && (\ell \ge 2)
\end{aligned}$$

any deterministic decision tree of depth at most $q_{\ell,m}$ can find the fixed point of $\widehat{F} \sim \widehat{\mathcal{F}}$ with probability at most $3m^{-c_0 - \frac{1}{8\ell}} = \varepsilon_{\ell,m}$. Note that any decision tree that finds a fixed point of F can be simulated by a decision tree that finds a fixed point of $\widehat{F}$ with the same number of queries, and F and $\widehat{F}$ have the same unique fixed point. The same probability bound also holds for $F \sim \mathcal{F}_{\ell,m}$. □

Acknowledgment and AI disclosure

Xi Chen and Yuhao Li were supported by NSF grants CCF-2106429 and CCF-2107187, and a grant from the AI Security Institute. Ruiquan Gao and Aviad Rubinstein were supported by a David and Lucile Packard Fellowship. Mihalis Yannakakis was supported by NSF grants CCF-2107187, CCF-2212233, and CCF-2332922.

The authors found both results via extensive discussions with GPT-5.6 Sol. The authors wrote the paper with the assistance of GPT-6 Astra and take full responsibility for all the content.

References

[BPR25] Simina Brânzei, Reed C. Phillips, and Nicholas J. Recker. Tarski lower bounds from multi-dimensional herringbones. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, volume 353 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:12, 2025. 1

[CC77] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL 1977)*, pages 238–252, 1977. 1

[CJK+17] Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2017)*, pages 252–263, 2017. 1

[CL22] Xi Chen and Yuhao Li. Improved upper bounds for finding Tarski fixed points. In *Proceedings of the 23rd ACM Conference on Economics and Computation (EC 2022)*, pages 1108–1118, 2022. 1, 2.1, 2.1

[CLY23] Xi Chen, Yuhao Li, and Mihalis Yannakakis. Reducing Tarski to unique Tarski (in the black-box model). In *Proceedings of the 38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 21:1–21:23, 2023. 1, 2.1, 4

[CLY24] Xi Chen, Yuhao Li, and Mihalis Yannakakis. Computing a fixed point of contraction maps in polynomial queries. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC 2024)*, pages 1364–1373, 2024. 1

[CLY26] Xi Chen, Yuhao Li, and Mihalis Yannakakis. The mystery deepens: On the query complexity of Tarski fixed points. In *Proceedings of the 67th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2026)*, 2026. To appear. 1, 1, 2.1, 2.1, 2.1, 4

[Con92] Anne Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992. 1

[DGK+17] Jérôme Dohrau, Bernd Gärtner, Manuel Kohler, Jiří Matoušek, and Emo Welzl. ARRIVAL: A zero-player graph game in NP ∩ coNP. In *A journey through discrete mathematics: A tribute to Jiří Matoušek*, pages 367–374. Springer, 2017. 1

[DQY11] Chuangyin Dang, Qi Qi, and Yinyu Ye. Computational models and complexities of Tarski's fixed points. Technical report, Stanford University, 2011. 1

[EJ91] E. Allen Emerson and Charanjit S. Jutla. Tree automata, μ-calculus and determinacy (extended abstract). In *Proceedings of the 32nd Annual IEEE Symposium on Foundations of Computer Science (FOCS 1991)*, pages 368–377, 1991. 1

[EJS93] E. Allen Emerson, Charanjit S. Jutla, and A. Prasad Sistla. On model-checking for fragments of μ-calculus. In *Proceedings of the 5th International Conference on Computer Aided Verification (CAV 1993)*, volume 697 of *Lecture Notes in Computer Science (LNCS)*, pages 385–396, 1993. 1

[EN01] Larry Eisenberg and Thomas H. Noe. Systemic risk in financial systems. *Management Science*, 47(2):236–249, 2001. 1

[EPRY20] Kousha Etessami, Christos H. Papadimitriou, Aviad Rubinstein, and Mihalis Yannakakis. Tarski's theorem, supermodular games, and the complexity of equilibria. In *Proceedings of the 11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:19, 2020. 1, 3, 3.1, 13, 5

[FPS22] John Fearnley, Dömötör Pálvölgyi, and Rahul Savani. A faster algorithm for finding Tarski fixed points. *ACM Transactions on Algorithms*, 18(3):23:1–23:23, 2022. 1, 2.1, 2.1

[GHH21] Bernd Gärtner, Sebastian Haslebacher, and Hung P. Hoang. A subexponential algorithm for ARRIVAL. In *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 69:1–69:14, 2021. 1

[GHJ+24] Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. Separations in proof complexity and TFNP. *Journal of the ACM*, 71(4):26:1–26:45, 2024. 1

[HL26] Sebastian Haslebacher and Jonas Lill. A levelset algorithm for 3D-Tarski. In *Proceedings of the SIAM Symposium on Simplicity in Algorithms (SOSA 2026)*, pages 56–64, 2026. 1

[HLSW25] Sebastian Haslebacher, Jonas Lill, Patrick Schnider, and Simon Weber. Query-efficient fixpoints of ℓ_p-contractions. In *Proceedings of the 66th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2025)*, pages 2037–2053, 2025. 1

[Koz83] Dexter Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983. 1

[LMYZ26] Tongyang Li, Weiran Ma, Ziyi Yang, and Xingyu Zhao. Quantum query complexity of finding a Tarski fixed point on a high-dimensional grid. *arXiv preprint arXiv:2609.03802*, 2026. 1

[MR90] Paul Milgrom and John Roberts. Rationalizability, learning, and equilibrium in games with strategic complementarities. *Econometrica*, 58(6):1255–1277, 1990. 1

[Ope26] OpenAI. Turn-Based Stochastic Mean-Payoff Games in Deterministic Quasipolynomial Time. OpenAI Math Release preprint OAI:Turn-Based-Stochastic-Mean-Payoff-Games-in-Deterministic-Quasipolynomial-Time-October-5-2026, 2026. 1

[Phi26] Reed Phillips. The quantum query complexity of finding a Tarski fixed point on the 2D grid. *arXiv preprint arXiv:2604.08223*, 2026. 1

[SS71] Dana S. Scott and Christopher Strachey. Toward a mathematical semantics for computer languages. Technical Monograph PRG06, Oxford University Computing Laboratory, 1971. 1

[Tar55] Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955. 1

[Top79] Donald M. Topkis. Equilibrium points in nonzero-sum n-person submodular games. *SIAM Journal on Control and Optimization*, 17(6):773–787, 1979. 1

[Top98] Donald M. Topkis. *Supermodularity and complementarity*. Princeton University Press, 1998. 1

[Wig19] Avi Wigderson. *Mathematics and computation: A theory revolutionizing technology and science*. Princeton University Press, 2019. 1

[ZP96] Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1–2):343–359, 1996. 1

A Proof of Theorem 2

We prove Theorem 2 by solving the recurrence in Theorem 5 and applying Theorem 4 and lemma 1. The same argument allows the factor 5^k to be replaced by C^k for any fixed $C > 4$; for simplicity, we take $C = 5$.

Proof of Theorem 2. Set $e(r) = \lceil \log((r+3)/3) \rceil$ for integers $r \geq 0$. We claim that there is a constant A such that

$$\textsc{SafeTarski}^*(n, r) \leq A 5^r (\log n)^{e(r)}.$$

For $r \geq 2$, write $m = \lfloor r/2 \rfloor - 1$. Then $1 + e(m) = e(r)$: both sides are at most an integer $j \geq 1$ exactly when $r \leq 3 \cdot 2^j - 3$. Since $e(0) = 0$ and $e(1) = 1$, the base cases $\textsc{SafeTarski}^*(n, 0) = 1$ and $\textsc{SafeTarski}^*(n, 1) = O(\log n)$, together with (3), give $\textsc{SafeTarski}^*(n, r) = O((\log n)^{e(r)})$ for each fixed r.

Choose an absolute constant $D > 0$ such that (3) holds with multiplier $Dr2^r \log n$. Then

$$Dr\, 2^r 5^{m-r} \leq \frac{Dr}{5} \left(\frac{2}{\sqrt{5}} \right)^r \longrightarrow 0.$$

Thus $Dr\, 2^r 5^m \leq 5^r$ for all sufficiently large r. Choose A to cover the finitely many smaller dimensions. Induction using (3) now gives

$$\textsc{SafeTarski}^*(n, r) \leq Dr\, 2^r \log n \cdot A 5^m (\log n)^{e(m)} \leq A 5^r (\log n)^{e(r)},$$

proving the claim.

For $k \geq 3$, Lemma 1 gives

$$\textsc{Tarski}(n,k) = O((\log n)\textsc{Tarski}^*(n,k-1))\,. \tag{9}$$

Combining (9) with Theorem 4 and the claim gives

$$\textsc{Tarski}(n,k) = O\Big(5^{k-1}(\log n)^{1+e(k-1)}\Big) = O\Big(5^k(\log n)^{1+\lceil\log((k+2)/3)\rceil}\Big)\,.$$

Finally, let $j = \lceil \log k \rceil \geq 2$. Since $k+2 \leq 2^j + 2 \leq 3 \cdot 2^{j-1}$, we have

$$1 + \left\lceil \log \frac{k+2}{3} \right\rceil \leq j = \lceil \log k \rceil.$$

This yields the simplified bound $\textsc{Tarski}(n,k) = O\big(5^k(\log n)^{\lceil \log k \rceil}\big)$, completing the proof. □