An Optimal Algorithm for Certifying Monotone Functions

Meghal Gupta ∗
Microsoft Research

Naren Sarayu Manoj †
Toyota Technological Institute Chicago

April 5, 2022

Abstract
Given query access to a monotone function \( f: \{0,1\}^n \to \{0,1\} \) with certificate complexity \( C(f) \) and an input \( x^* \), we design an algorithm that outputs a size-\( C(f) \) subset of \( x^* \) certifying the value of \( f(x^*) \). Our algorithm makes \( O(C(f) \cdot \log n) \) queries to \( f \), which matches the information-theoretic lower bound for this problem and resolves the concrete open question posed in the STOC ’22 paper of Blanc, Koch, Lange, and Tan [BKL T22].

We extend this result to an algorithm that finds a size-\( 2C(f) \) certificate for a real-valued monotone function with \( O(C(f) \cdot \log n) \) queries. We also complement our algorithms with a hardness result, in which we show that finding the shortest possible certificate in \( x^* \) may require \( \Omega\left(\left(\frac{n}{C(f)}\right)\right) \) queries in the worst case.

1 Introduction
Given a function \( f: \{0,1\}^n \to \mathbb{D} \) for some output domain \( \mathbb{D} \) and an input \( x^* \), is there a short proof for why \( f(x^*) \) takes on the value it does? This natural question motivates the notion of certificate complexity in complexity theory. Loosely speaking, a certificate for \( f(x^*) = y \) is a subset of the bits of \( x^* \) that “fixes” the value of \( f(x^*) \). In other words, every input \( x \) that agrees with \( x^* \) on the bits in the certificate will satisfy \( f(x) = f(x^*) \). Besides being a quantity of interest in complexity theory and in the analysis of Boolean functions, certificate complexity has a natural interpretation in the context of explainable AI. Here, the practitioner aims to find simple properties of a given input that explain a classifier’s prediction on the input. We formalize the notion of a certificate in Definition 1.1.

Definition 1.1 (Certificate (see, e.g., [AB09])). Let \( x|_S \) denote the substring of \( x \) in the coordinates of \( S \).
For a function \( f: \{0,1\}^n \to \mathbb{D} \) and an input \( x^* \in \{0,1\}^n \), we say a set \( S \subseteq [n] \) is a certificate if for all \( y \in \{0,1\}^n \) such that \( x^*|_S = y|_S \), we have \( f(x^*) = f(y) \).

We use Definition 1.1 to define the certificate complexity of a function \( f \).

Definition 1.2 (Certificate complexity (see, e.g., [AB09])). For any function \( f: \{0,1\}^n \to \mathbb{D} \) and \( x \in \{0,1\}^n \), we let \( C(f, x) \) be the smallest integer such that there exists a \( C(f, x) \)-sized certificate for \( f(x) = j \). We now let the certificate complexity of \( f \) be \( \max_{x \in \{0,1\}^n} C(f, x) \).

∗E-mail: meghal@mit.edu
†E-mail: nsm@ttic.edu.
A natural follow-up question from Definition 1.2 is whether a short certificate can be found in a given input if we know that all inputs have a short certificate. The following problem, posed and studied in the STOC ’22 paper of [BKL22], formalizes this question.

**Problem.** Given queries to a function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) with certificate complexity \( C(f) \) and an input \( x^* \), output a size-\( C(f) \) certificate for \( f \)'s value on \( x^* \).

The main result of [BKL22] is an algorithm for the case where \( f \) is monotone and the output range \( \mathbb{D} = \{0, 1\} \). The authors design a randomized algorithm that makes at most \( O(C(f)^{8} \cdot \log n) \) queries using a novel connection to threshold phenomena. Furthermore, [BKL22] show that \( \Omega(C(f) \cdot \log n) \) queries for the certification problem are necessary in the worst-case. The authors identify closing this gap as a concrete direction for future work.

### 1.1 Our Results

Our main result is a simple, deterministic algorithm that makes \( O(C(f) \cdot \log n) \) queries to find a size-\( C(f) \) certificate for any monotone binary-valued function \( f \) and input \( x^* \). This completely resolves the aforementioned open question from [BKL22]. Formally, we have Theorem 1.3.

**Theorem 1.3.** Given query access to a monotone function \( f : \{0, 1\}^n \rightarrow \{0, 1\} \) and an input \( x^* \), there exists an algorithm that makes \( O(C(f) \cdot \log n) \) queries to \( f \) and outputs a size-\( C(f) \) subset \( S \) corresponding to a subset of indices of \( x^* \) certifying the value of \( f(x^*) \).

We can extend our result to obtain as a simple corollary an algorithm that finds a size-\( 2C(f) \) certificate for any monotone real-valued function \( f \) and input \( x^* \). Specifically, we have Theorem 1.4.

**Theorem 1.4.** Given query access to a monotone function \( f : \{0, 1\}^n \rightarrow \mathbb{R} \) and an input \( x^* \), there exists an algorithm that makes \( O(C(f) \cdot \log n) \) queries to \( f \) and outputs a size-\( 2C(f) \) subset \( S \) corresponding to a subset of indices of \( x^* \) certifying the value of \( f(x^*) \).

The careful reader might also wonder why we are only looking for a certificate of size-\( C(f) \) on the input \( x^* \) – by definition, the shortest certificate on a fixed input \( x^* \) is size-\( C(f, x^*) \). We show that finding a certificate of length \( C(f, x^*) \) may require far more queries than simply finding one of length \( C(f) \). In particular, in the case where \( C(f, x^*) = n/2 \), it may require exponentially many queries. Moreover, our result matches the trivial upper bound provided by an algorithm which simply queries all \( C(f, x^*) \)-size certificates. See Theorem 1.5.

**Theorem 1.5.** For any \( k \), for any (randomized) algorithm that queries a given function, there exists a function \( f \) and input \( x^* \) such that \( k = C(f, x^*) \), and the algorithm must make at least \( \frac{1}{2} \binom{n}{k} \) queries to determine a certificate with probability \( > 1/2 \).

### 1.2 Related Work

We derive our setting and problem statements from the work of [BKL22]. The authors of [BKL22] formally propose the problem of certifying a monotone function \( f \) on an input \( x^* \) and provide an algorithm for doing so, as mentioned earlier. They also look at the certification question for a general (non-monotone) function \( f \). Here, they show \( \Omega(2^{C(f)} + C(f) \cdot \log n) \) queries are necessary, and
queries suffice with high probability. Closing this gap remains an interesting open direction.

Before the work of [BKLT22], Angluin (see [Ang88]) gave a local search algorithm that can be used to certify a monotone function $f$ on an input $x^*$ with query complexity $O(n)$. See Appendix C in [BKLT22] for a detailed exposition and proof of correctness of Angluin’s algorithm.

2 Preliminaries

Notation In this work, we use the following notation.

• We denote the set $\{x \in \mathbb{Z}_{\geq 0} : 1 \leq x \leq n\}$ as $[n]$. In an abuse of notation, let $[0] = \emptyset$.

• For a set $S \subseteq [n]$, we write $x_S$ to be the indicator vector for $S$; i.e., $x_S$ is such that $x_i = 1 \{i \in S\}$, for all $i \in [n]$. Additionally, we write $x|_S$ to be the substring of $x$ in the coordinates of $S$. Specifically, we have $x|_S = \{(i, x_i) \text{ for all } i \in S\}$.

• Let $\mathbb{1}^n$ denote the all-1s vector in $n$ dimensions.

• For a vector $x \in \{0, 1\}^n$, we denote $S_x$ to be $\{i : x_i = 1\}$.

In our work, it is helpful to distinguish a minimal certificate from a general certificate.

Definition 2.1 (Minimal Certificate). For a given function $f$, we say a certificate $S \subseteq [n]$ is minimal if for all $a \in S$, we have that $S \setminus a$ is not a certificate for $f(x)$. If $f$ is monotone, this is equivalent to requiring that for all $A \subset S$, we have $f(x|_A) \neq f(x|_S)$.

Finally, we note the information-theoretic lower bound from [BKLT22] on the query complexity of any algorithm used to certify $f(x)$ for a monotone function $f$.

Lemma 2.2 (Claim 1.2 in [BKLT22]). For any $c < 1$ and any $k \leq l \leq n^c$, let $A$ be an algorithm which, given query access to a monotone function $f : \{0, 1\}^n \to \{0, 1\}$ with certificate complexity $C(f)$ and an input $x^*$, returns a size-$l$ certificate for $f$’s value on $x^*$ with high probability. The query complexity of $A$ must be $\Omega(k \log n)$.

3 Our Algorithm to Certify a Binary Monotone Function

We first restate the problem.

Problem. Given query access to a monotone function $f : \{0, 1\}^n \to \{0, 1\}$ with certificate complexity $C(f)$ and an input $x^*$, output a size-$C(f)$ certificate for $f$’s value on $x^*$.

3.1 Overview of Our Algorithm

We informally describe our algorithm. Without loss of generality, we let $x^*$ be such that $f(x^*) = 1$. A valid certificate is any subset $S \subseteq S_{x^*}$ of indices such that $f(x_S) = 1$.

We add elements into our certificate $A$ one-by-one. To do this, we simply iterate the following steps until $A$ is a valid certificate:

1. Find the smallest $s \in S_{x^*}$ such that including all $i \leq s \in S_{x^*}$ in the certificate, along with elements already in $A$, yields a valid certificate.
2. Add \( s \) to \( A \).

Because the function is monotone, \( s \) can be found through binary search at each step. Moreover, observe that removing any one element from \( A \) no longer yields a valid certificate; thus, as we will show in Lemma 3.3, the output certificate is length at most \( C(f) \). This also implies the algorithm makes a total of \( O(C(f) \cdot \log n) \) queries.

### 3.2 Formal Description of Our Algorithm

We state our algorithm formally. In our algorithm description and analysis, we assume without loss of generality that \( f(x^*) = 1 \). We can make this assumption since if \( f(x^*) = 0 \), we can instead run the algorithm making queries to \( g(x) := 1 - f(1^n - x) \), which is a monotone function with \( g(x^*) = 1 \).

**Definition 3.1 (search).** The procedure \( \text{search}(f, A, S) \) acts on a monotone function \( f \in \{0, 1\}^n \rightarrow \{0, 1\} \), and two sets \( A, S \subseteq \mathbb{Z} \). If \( f(A) = 1 \) or \( f(A \cup S) = 0 \), it outputs ERROR. Else, it outputs the smallest \( s \in S \) for which \( f(A \cup \{s\} \cap S) = 1 \). The function proceeds using binary search, which can be done because \( f \) is monotone.

**Algorithm 1 : Algorithm to Certify a Binary Monotone Function Where \( f(x^*) = 1 \)**

1. **Input:** Query access to a function \( f: \{0, 1\}^n \rightarrow \{0, 1\} \) and point \( x^* \) for which \( f(x^*) = 1 \).
2. Initialize the sets \( A \leftarrow \emptyset \) and \( S \leftarrow S_{x^*} \).
3. Run the following procedure until \( f(A) = 1 \):
   (a) Set \( s \leftarrow \text{search}(f, A, S) \).
   (b) Add \( s \) to \( A \).
   (c) Set \( S \leftarrow S \cap [s - 1] \)
4. **Output:** \( A \).

### 3.3 Analysis

**Theorem 3.2.** The Algorithm in 3.2 outputs a certificate of length at most \( C(f) \) for \( f \) on \( x^* \) and makes at most \( O(C(f) \cdot \log n) \) queries.

We break the proof down into a series of lemmas.

**Lemma 3.3.** If \( S \) is a minimal certificate, then \( |S| \leq C(f) \).

**Proof.** Consider the shortest certificate \( C \) for the input \( x_S \). We must have \( f(C) = f(x_S) \), and \( |C| \leq C(f) \). The fact that \( |C| < |S| \) and \( f(C) = f(x_S) \) contradicts that \( S \) is minimal. \( \square \)

**Lemma 3.4.** The Algorithm in 3.2 never outputs ERROR.

**Proof.** If the algorithm outputs ERROR, it must be in Step 3a. By definition, an error occurs if \( f(A) = 1 \) or \( f(A \cup S) = 0 \). The former cannot be true because the algorithm checks this exact condition in Step 3. The latter cannot be true because:
• If this is the first iteration of Step 3, \( A \cup S = S_{x^*} \), which means \( f(x_{A \cup S_{x^*}}) = f(x^*) = 1 \).

• Else, in the previous iteration of Step 3a (let the values of \( A, S, s \) at that step be \( A', S', s' \) respectively), it must have been the case that \( f(x_{A' \cup (S' \cap [s'])}) = 1 \). Note that \( A' \cup (S' \cap [s']) = A \cup (S' \cap [s' - 1]) = A \cup S \), and so \( f(x_{A \cup S}) = 1 \).

Lemma 3.5. If the Algorithm in 3.2 terminates, it outputs a minimal certificate for \( f \) on \( x^* \).

Proof. It must be the case that \( f(A) = 1 \); otherwise, we could not have left Step 3. Consider any \( s \in A \) and we will show that \( f(A \setminus s) = 0 \).

At the iteration of Step 3 where \( s \) was added to \( A \) (let the temporary certificate \( A \) at the start of that step be \( A_s \)), it must be the case that \( f(x_{(S \cap [s]) \cup A_s}) = 1 \) but \( f(x_{(S \cap [s - 1]) \cup A_s}) = 0 \). All future elements that are added to create the final certificate \( A \) must be a subset of \( S \cap [s - 1] \) (where \( S \) is being referenced from the current iteration of Step 3). Therefore, \( A \setminus S \subseteq (S \cap [s - 1]) \cup A_s \), and therefore \( f(A \setminus s) = 0 \).

Lemma 3.6. The Algorithm in 3.2 terminates, making at most \( O(C(f) \log n) \) queries.

Proof. Observe that in every iteration of the main loop, we add exactly one element to \( A \). By Lemma 3.3, there are at most \( C(f) \) coordinates in the output \( A \). Hence, we run the main loop at most \( C(f) \) times.

Next, \( \text{search}(f, A, S) \) is a binary search over a domain of size \( |S| \leq |S_{x^*}| \leq n \). Therefore, \( \text{search}(f, A, S) \) uses at most \( \log n \) queries.

Finally, the check \( f(x^*_A) = 1 \) costs 1 query, and this runs at the beginning of every iteration of the loop. In total, we make at most \( C(f) \cdot (\log n + 1) \) queries, as desired.

Combining these lemmas finishes the proof of Theorem 3.2.

4 Extension to Real-Valued Functions

In this section, we prove the following corollary of our main result wherein the output domain is \( \mathbb{R} \) instead of \( \{0, 1\} \).

Corollary 4.1. There exists an algorithm that, given an input \( x^* \) and query access to a monotone \( f: \{0, 1\}^n \to \mathbb{R} \), makes \( O(C(f) \cdot \log n) \) queries to \( f \) and outputs a size-\( 2 \cdot C(f) \) certificate for \( f(x^*) \).

4.1 Our Algorithm

We begin with two necessary definitions.

Definition 4.2 (binary\_cert\((f, x^*)\)). The procedure \( \text{binary\_cert}(f, x^*) \) is given query access to function \( f: \{0, 1\}^n \to \{0, 1\} \) and an input \( x^* \), runs our algorithm from Section 3, and outputs a size-\( C(f) \) certificate for \( f(x^*) \).
Definition 4.3 \((g_{0,f,x^*}(x), g_{1,f,x^*}(x))\). Let \(b \in \{0,1\}\), \(f : \{0,1\}^n \to \mathbb{R}\) and \(x^* \in \{0,1\}^n\). The function \(g_{b,f,x^*} : \{0,1\}^n \to \{0,1\}\) is defined as follows:

\[
g_b(x) = \begin{cases} 
0 & f(x) < f(x^*) \\
1 & f(x) > f(x^*) \\
b & f(x) = f(x^*)
\end{cases}
\]

We will abbreviate \(g_{b,f,x^*} : \{0,1\}^n \to \{0,1\}\) as \(g_b\) when \(f\) and \(x^*\) are clear.

Algorithm 2: Algorithm to Certify a Real-Valued Monotone Function

1. **Input:** \(f, x^*\).
2. Set \(C_0 \leftarrow \text{binary}_\text{cert}(g_0, x^*)\).
3. Set \(C_1 \leftarrow \text{binary}_\text{cert}(g_1, x^*)\).
4. **Output:** \(C_0 \cup C_1\).

4.2 Analysis

Theorem 4.4. The Algorithm in 4.1 outputs a certificate for \(f\) of length at most \(2C(f)\) and makes at most \(O(C(f) \log n)\) queries.

We break the proof into a series of lemmas. Call the output \(A\).

Lemma 4.5. \(A\) is a valid certificate for \(f\) on \(x^*\).

Proof. For any input \(y\) such that \(y|_A = x^*|_A\), we must have \(g_b(y) = g_b(x^*)\) for both \(b = 0, 1\). Notice that both of the following hold:

\[
g_0(y) = g_0(x^*) \text{ implying } f(y) \leq f(x^*) \\
g_1(y) = g_1(x^*) \text{ implying } f(y) \geq f(x^*)
\]

Hence, we have \(f(y) = f(x^*)\). \(\square\)

Lemma 4.6. \(|A| \leq 2C(f)\).

Proof. It suffices to show that \(C(f) \geq C(g_b)\) for \(b \in \{0,1\}\). We will show that any certificate \(B\) for \(f\) on \(x\) is also a certificate for \(g_b\). For all \(y, y'\) with \(y|_C = y'|_C\), we have \(f(y) = f(y')\), but this implies by definition that \(g_b(y) = g_b(y')\). Hence, \(B\) is also a certificate for \(g_b\). \(\square\)

Combining these lemmas concludes the proof of Theorem 4.4.
5 Finding the Shortest Certificate for a Monotone Function

In this section, we show that there exists a family of instances on which the problem of finding the shortest certificate for a binary-valued \( f \) on an input \( x^* \) (denoted \( k := C(f, x^*) \)) requires at least \( \Omega \left( \binom{n}{k} \right) \) queries.

Notice that this result is essentially optimal: for any function \( f \), any input \( x^* \) and \( k = C(f, x^*) \), \( O \left( \binom{n}{k} \right) \) suffice to find a size-\( k \) certificate. Assuming \( f(x^*) = 1 \), the algorithm can simply query \( f(x_S) \) for all subsets \( S \) of size \( k \) and check if each one of them is a certificate.

**Definition 5.1** \( (F_k) \). We define the set of \( k \)-indicator functions, denoted \( F_k \) as follows.

Let \( f_P : \{0,1\}^n \rightarrow \{0,1\} \) for some \( P \subseteq [n] \) be defined as follows:

\[
f_P(x) = \begin{cases} 
0 & |S_x| < k \\
1 & |S_x| > k \\
0 & |S_x| = k, P \neq S_x \\
1 & |S_x| = k, P = S_x 
\end{cases}
\]

Finally, let \( F_k = \{f_P : |P| = k\} \).

**Lemma 5.2.** Every function \( f \in F_k \) has \( C(f, 1^n) = k \).

**Proof.** It is easy to see that every function in \( F_k \) is monotone for all \( k \).

Let \( P \subset [n] \) be such that \( f_P = f \). Observe that \( |P| = k \). Next, notice that \( f(1_P^n) = f(1^n) = 1 \). This implies that \( C(f, 1^n) \leq |P| = k \). Finally, consider any \( S \subset [n] \) such that \( |S| < k \). Note that for \( x = 1^n_S \), we have \( |S_x| < k \), so \( f_P(x) = 0 \). Thus, we have \( C(f, 1^n) \geq k \), and we're done. \( \square \)

**Theorem 5.3.** For any \( k \in [n - 1] \) and (randomized) algorithm \( A \), there exists a function \( f : \{0,1\}^n \rightarrow \{0,1\} \) and input \( x^* \) with \( C(f, x^*) = k \) such that \( A \) must make at least \( \frac{1}{2} \cdot \binom{n}{k} \) queries to \( f \) to find the size-\( k \) certificate with probability \( \geq \frac{1}{2} \).

**Proof.** Fix an arbitrary \( k \in [n-1] \). We will show that some function \( f \in F_k \) takes \( \geq \frac{1}{2} \cdot \binom{n}{k} \) queries to certify on the input \( 1^n \). Note that \( C(f, 1^n) = k \). Additionally, observe that any randomized algorithm to find a certificate for \( f(1^n) = 1 \) can be converted to one that only makes queries \( x \) satisfying \( |S_x| = k \).

Let \( X = \{x \in \{0,1\}^n : |S_x| = k\} \). Notice that \( |X| = \binom{n}{k} \). Any randomized algorithm for finding the single \( x \in X \) such that \( f(x) = 1 \) can be viewed as one that samples a permutation from some distribution over permutations of \( X \) and makes queries to \( f \) in the order determined by the permutation until the algorithm encounters the \( x \in X \) for which \( f(x) = 1 \). This is because query \( i \) only depends on the values of the queries \( 1, \ldots, i-1 \), and not their responses – in particular, the responses to queries \( 1, \ldots, i-1 \) are all 0 if the algorithm has not terminated prior to issuing query \( i \). With this interpretation in mind, fix some distribution of permutations of \( X \); call this distribution \( \mathcal{P} \).

For each \( x \in X \), consider \( \Pr_{P \in \mathcal{P}} [P^{-1}(x) \leq |X|/2] \) where \( P^{-1}(x) \) is the index of element \( X \). Let \( \mu(P) \) denote the probability that a random permutation drawn from \( \mathcal{P} \) is \( P \), and observe the
following manipulations:

\[
\sum_{x \in X} \sum_{P \in \mathcal{P}} \Pr_{X \in \mathcal{P}} \{ P^{-1}(x) \leq |X|/2 \} = \sum_{x \in X} \sum_{P} \mu(P) \cdot 1 \{ P^{-1}(x) \leq |X|/2 \} \\
= \sum_{P} \mu(P) \cdot \sum_{x \in X} 1 \{ P^{-1}(x) \leq |X|/2 \} \\
= \sum_{P} \mu(P) \cdot \frac{1}{2} \cdot |X| = \frac{1}{2} \cdot |X|
\]

Thus, there exists at least one \( x \in X \) for which \( \Pr_{P \in \mathcal{P}} \{ P^{-1}(x) \leq |X|/2 \} \leq 1/2 \). It follows that the algorithm does not find a sized-\( k \) subset of \( \mathbb{1}^n \) certifying \( f(\mathbb{1}^n) = 1 \) with probability \( > 1/2 \) without making at least \( \frac{1}{2} \binom{n}{k} \) queries.

\[
\square
\]

References

[Ang88] Dana Angluin. Queries and concept learning. *Machine learning*, 2(4):319–342, 1988.

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, USA, 1st edition, 2009. ISBN: 0521424267.

[BKLT22] Guy Blanc, Caleb Koch, Jane Lange, and Li-Yang Tan. The query complexity of certification, 2022. DOI: 10.48550/ARXIV.2201.07736. URL: https://arxiv.org/abs/2201.07736.

[ODo21] Ryan O’Donnell. Analysis of boolean functions. *arXiv preprint arXiv:2105.10386*, 2021.