IACR Communications in Cryptology
ISSN 3006-5496, Vol. 2, No. 1, 18 pages.

https://doi.org/10.62056/a3txom2hd

Adversarially Robust Bloom Filters: Monotonicity and Betting

Chen Lotan and Moni Naor

Weizmann Institute of Science, Department of Computer Science and Applied Mathematics, Rehovot, Israel

Abstract.
A Bloom filter is a probabilistic data structure designed to provide a compact representation of a set S of elements from a large universe U. The trade-off for this succinctness is allowing some errors. The Bloom filter efficiently answers membership queries: given any query x, if $x \in S$, it must answer 'Yes'; if $x \notin S$, it should answer 'Yes' only with a probability of at most ε.
Traditionally, the error probability of the Bloom filter is analyzed under the assumption that the query is independent of its internal randomness. However, Naor and Yogev (Crypto 2015) focused on the behavior of this data structure in adversarial settings where the adversary may choose the queries adaptively. One particular challenge in this direction is to define rigorously the robustness of Bloom filters in this model.
In this work, we continue investigating the definitions of success of the adaptive adversary. Specifically, we focus on two notions proposed by Naor and Oved (TCC 2022) and examine the relationships between them. In particular, we highlight the notion of Bet-or-Pass as being stronger than others, such as Monotone-Test Resilience.

Keywords: Bloom filters · adversarial settings · pseudorandomness

1 Introduction

Consider a long-lived system where at each step there is a certain probability of failure. Suppose that if the inputs to the system are chosen ahead of time, this probability of failure is non-negligible but can be bounded by some parameter ε. Now, suppose that instead we have an *adaptive adversary* that chooses the inputs based on the reaction of the system in previous rounds. How should we define the bound on the success of such an adaptive adversary when even the static adversary has some non-negligible chance of success?

We focus on the non-negligible case since in the *negligible* case, we expect no failures at all, and the natural definition for the adaptive case is simply that no failure occurs. In the *non-negligible* case, we aim to ensure that the adversary does not perform better than chance at each step. This paper specifically examines this scenario within the context of Bloom filters.

A Bloom filter is a probabilistic data structure designed to provide a compact representation of a set S of elements from a large universe U. The trade-off for this succinctness is allowing some errors. The Bloom filter efficiently answers membership queries: given any

Research supported in part by grants from the Israel Science Foundation (no. 2686/20), by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness and by the Israeli Council for Higher Education (CHE) via the Weizmann Data Science Research Center

E-mail: chen.lotan@weizmann.ac.il (Chen Lotan), moni.naor@weizmann.ac.il (Moni Naor)

Received: 2025-01-13 Accepted: 2025-03-11

query x, if $x \in S$ it must answer 'Yes'; if $x \notin S$ it should answer 'Yes' only with probability at most ε. Those errors are called *false positives* (FPs).

The construction of a Bloom filter requires only a small memory size compared to storing S precisely. Moreover, they have a very fast query time. Those properties make Bloom filters extremely practical and useful in various areas (see for instance [BM03, TRL12, NPB21]). Note that we use 'Bloom filter' in this work as a synonym for a data structure that implements *approximate set membership*, not necessarily to the original implementation of such data structures suggested by Bloom [Blo70].

Usually, the correctness of a Bloom filter is analyzed under the assumption that we first fix a query x and then compute the error probability over the internal randomness of the Bloom filter. However, this analysis might not suffice in many scenarios, among them the case of a series of queries that are chosen adaptively, based on previous responses. This raises the need to analyze the Bloom filter in a more robust model. Recent papers suggested different robustness definitions; in this work we contribute to this ongoing exploration by examining the relations between different notions of robustness.

Naor and Yogev [NY15] initiated the analysis of Bloom filters that are resilient against adaptive choices, where the adversary may choose queries based on the answer to the previous ones (but doesn't have access to the internal representation). They came up with a notion (later called Always-Bet) and constructions satisfying it, as well as impossibility results showing the need for computational assumptions if the number of queries is unbounded. Naor and Oved [NO22] refined the resiliency notion and came up with two notions of robustness and explored the relationships between them. The robustness test is defined as a game with an adversary, the adversary chooses the set S and adaptively queries the Bloom filter. The adversary's objective varies depending on the specific test being conducted.

The key idea is that a "robust" Bloom filter will behave like a truly unpredictable biased coin; that is, each query is false positive with probability at most ε regardless of the result of previous queries. This idea is formalized in the definition of the *Monotone test* introduced in their work. Informally, this test checks whether the output of a sequence of adaptive queries can be distinguished from a biased random coin. We consider monotone distinguishers only since we are interested in preventing the cases where the adversary is able to increase the false positive rate, not decrease it. If a Bloom filter does not fail any monotone test, we say that it is Monotone test resilient.

The Monotone test presents a "continuous" challenge: this kind of test examines the false positive rate of the entire sequence of adaptive queries and looks for "anomalies". We also consider a one-time challenge test in which an adversary performs a series of adaptive queries, and her goal is to find one never-queried-before false positive. An example of such a test is the *Always-Bet (AB) test* first presented by Naor and Yogev [NY15]. They defined that following the adaptive queries, the adversary must output a never-queried before element x^*, which she thinks is a false positive. The adversary wins the game if x^* is a false positive. They wanted the probability of an adversary to win to be at most ε.

The *Bet-or-Pass (BP) test*, presented by Naor and Oved [NO22], extends the AB test. In this test we strengthen the adversary by allowing her to pass, meaning she does not have to provide any output. We measure her success by defining the adversary's profit: if she outputs (bets on) an element x^* which is indeed a false positive, she is rewarded; otherwise, she "pays". If she chooses to pass, her profit is zero. The payments are set so that a random guess with probability ε has an expected profit of 0. We say that an adversary makes the Bloom filter fail in the Bet-or-Pass test if her expected profit is noticeably larger than 0.

Naor and Oved proved that BP Test Resilience implies Monotone test resilience, indicating that the concept of Bet-or-Pass accurately captures the desired behavior of this data structure in adversarial settings. Additionally, the Bet-or-Pass notion is not too

strong; they were able to show that the construction from [NY15] satisfies this notion. Those results lead them to conclude that the BP test is a natural notion for robustness. Another result they achieved is that AB test resilience does not necessarily imply Monotone test resilience, highlighting the 'superiority' of the BP-test definition.

For more discussion of relevant previous work, see the Conclusions and Open Problems section.

Our Contributions: Naor and Oved's work left open the question of whether Monotone test resilience implies BP test resilience or not. In this work, we prove in Section 4 that the two notions are actually separable: we show a construction of a Bloom filter **B** that fails the BP-test but is resilient to any Monotone test. Furthermore, we show that Monotone test resilience does imply AB test resilience. These results strengthen the [NO22] conclusion regarding the Bet-or-Pass test, showing that this test gives us the strongest guarantee we can (currently) imagine.

2 Notation and Preliminaries

2.1 Probabilistic Statements

For random variable S and distribution $\mathcal{D}$ we use the notation $X \sim \mathcal{D}$ to denote that X is distributed as $\mathcal{D}$.

Negligible Function: We use the notation negl for any function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}^+$ satisfying that for every positive polynomial $p(\cdot)$ there is an N such that for all $n > N$ it holds that $\mathsf{negl}(n) < \frac{1}{p(n)}$. Such functions are called *negligible*.

We will compare the outcome of the t queries with that of a vector of independent Bernoulli trials:

Definition 1. [Bernoulli distribution on vectors] We denote by $\mathrm{Ber}_{\varepsilon,t}$ the distribution of vectors in $\{0,1\}^t$, where each entry is chosen independently of all the other and has probability ε of being 1.

We will consider monotone tests, i.e. those determined by monotone functions, defined as follows:

Definition 2 (Monotone function)**.** Let $t \in \mathbb{N}$. We say that a function $f\colon \{0,1\}^t \to \{0,1\}$ is monotone if for every pair of neighboring strings $Y, Y' \in \{0,1\}^t$ that are equal in all locations except one index $1 \le i \le t$ i.e. $Y_j = Y'_j$ for all $j \neq i$ and $Y_i = 0$ and $Y'_i = 1$, we have that $f(Y) = 1$ implies that $f(Y') = 1$.

A property of monotonicity we will use is that for any monotone $f\colon \{0,1\}^t \to \{0,1\}$ it holds that for all $0 \le \alpha \le \varepsilon \le 1$ we have that increasing the probability from α to ε in $\mathrm{Ber}_{\cdot,t}$ implies a non decrease in the probability that f is 1 (See Kalai and Safra Section 2.3 [KS06]). I.e.

$$\Pr_{Y \sim Ber_{\alpha,t}}[f(Y) = 1] \le \Pr_{Y \sim Ber_{\epsilon,t}}[f(Y) = 1].$$

Proof. Let f be a monotone boolean function and consider three independent vector Bernoulli r.v.: $Y \sim Ber_{\varepsilon,t}$, $X \sim Ber_{\alpha,t}$ and $Z \sim Ber_{\frac{\varepsilon-\alpha}{1-\alpha},t}$. The point is that the distributions of Y and $X \vee Z$ (the bit wise OR) are identical: for every entry $i = 1, \ldots, t$

we have:

$$\begin{aligned}\Pr[X_i \vee Z_i = 1] =& \Pr[X_i = 1] + \Pr[Z_i = 1] - \Pr[X_i = 1 \wedge Z_i = 1] \\ =& \alpha + \frac{\varepsilon - \alpha}{1 - \alpha} - \alpha \cdot \frac{\varepsilon - \alpha}{1 - \alpha} \\ =& \varepsilon \\ =& \Pr[Y_i = 1]\end{aligned}$$

and $\forall i \neq j$ $X_i \vee Z_i$ and $X_j \vee Z_j$ are independent (since X_i, X_j are independent and Z_i, Z_j are independent). Hence, $(X \text{ OR } Z) \sim Ber_{\varepsilon,t}$. Then,

$$\Pr[f(Y) = 1] = \Pr[f(X \text{ OR } Z) = 1] \geq \Pr[f(X) = 1 \text{ OR } f(Z) = 1] \geq \Pr[f(X) = 1]$$

□

We will be using the following concentration bound:

Theorem 1. ***Multiplicative Chernoff bound*** *(See Mitzenmacher and Upfal [MU05] Section 4.2.1): Suppose $Y_1, \ldots, Y_t$ are independent random variables taking values in $\{0, 1\}$. Let $Y = \sum_{i=1}^{t} Y_i$ be their sum and let $\mu = \mathbb{E}[Y]$ be the expected value of the sum. Then,*

- $\Pr[Y \geq (1 + a)\mu] < e^{-a^2\mu/(2+a)}$ *for any* $0 \leq a$
- $\Pr[Y \leq (1 - a)\mu] < e^{-a^2\mu/2}$ *for any* $0 < a < 1$

2.2 Bloom Filters Preliminaries

We consider a universe U of elements (where $|U| = u$), and a given subset $S \subset U$ of size n. We will focus on the case where the set is fixed throughout the lifetime of the Bloom filter as was done by Naor and Oved [NO22],. We think of a Bloom filter as a data structure $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2)$ composed of two algorithms, a setup algorithm, and a query-answering algorithm.

Definition 3 (Bloom Filter)**.** Let $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2)$ be a pair of probabilistic polynomial time algorithms such that $\mathbf{B}_1$ gets as input a set of elements $S \subset U$ and outputs a representation M, and $\mathbf{B}_2$ gets a representation M and a query element $x \in U$ and outputs an answer to the query. We say that $\mathbf{B}$ is an (n, ε)-Bloom filter if for any set $S \subset U$ of size n it holds that:

1. Completeness: For any $x \in S$: $\Pr[\mathbf{B}_2(\mathbf{B}_1(S), x) = 1] = 1$
2. Soundness: For any $x \notin S$: $\Pr[\mathbf{B}_2(\mathbf{B}_1(S), x) = 1] \leq \varepsilon$

Where the probabilities are over the internal randomness of the setup algorithm $\mathbf{B}_1$ and query algorithm $\mathbf{B}_2$.

Since our main focus is the separation of definitions by a counter-example, we can consider for simplicity a query algorithm that maintains the set representation unchanged.

When $x \notin S$ and $\mathbf{B}_2(\mathbf{B}_1(S), x) = 1$ we classify x as a *false positive*. The false positive rate of a Bloom filter is its main evaluation metric.

We consider a series of Bloom filters with varying values of n; in our analysis, we focus on Bloom filters with sufficiently large n.

Memory Requirements: An important parameter of Bloom filters is their memory requirement. We denote this by m, which represents the number of bits the Bloom filter needs to store its data. Essentially, we want m to be as small as possible while still effectively answering membership queries. Carter et al. [CFG+78] proved that for a Bloom filter handling sets of size n and with an error rate of ε, if $u > n^2/\varepsilon$, the minimum required memory is approximately $m \geq n \log \frac{1}{\varepsilon}$ bits. This relationship can be expressed as $\varepsilon \geq 2^{-\frac{m}{n}}$, defining what we call the *minimal error* of the Bloom filter.

A straightforward way to construct a robust Bloom filter is by storing the set S explicitly, thereby eliminating false positives for any adversary, but consuming significant memory and contradicting the primary goal of Bloom filters: to minimize memory usage. Our focus, however, is on developing a robust, non-trivial Bloom filter. In broad terms, a non-trivial Bloom filter is characterized by an ε that is neither close to 0 nor 1, and a large universe size in comparison to the available memory, making it impossible to store the set explicitly.

Definition 4 (Non-trivial Bloom filter (Def 2.6 in [NO22]))**.** Let $\mathbf{B}$ be an (n, ε)-Bloom filter for a universe $\mathcal{U}$ that uses m bits of memory. Let $\varepsilon_0 = 2^{-\frac{m}{n}}$ be the minimal error and let the universe size u be s.t. $u > n^2/\varepsilon_0$. We say that $\mathbf{B}$ is *non-trivial* if there exists polynomials $p_1(\cdot), p_2(\cdot)$ such that $\frac{1}{p_1(n)} < \varepsilon_0 \leq \varepsilon < 1 - \frac{1}{p_2(n)}$.[1]

3 Robustness Definitions of Bloom filters

3.1 Modeling the Adversary

In Definition 3 above for Bloom filters, the probability of failure is for a single given element x and is taken over the randomness of $\mathbf{B}$. We want to explore the robustness of Bloom filters against stronger adversaries who have the power to make multiple queries $x_1, \ldots, x_t$ that are not fixed but chosen adaptively (i.e., the adversary can see the responses of previous queries before choosing the next one). We introduce a *security parameter* λ provided as input to both the setup algorithm $\mathbf{B}_1$ and the adversary. Now the running time of the Bloom filter should be polynomial in λ.

To formalize this adversary model we use Naor and Oved's definition for the game

$$\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$$

where λ is the security parameter. In this game, we consider an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, where $\mathcal{A}_1$ chooses the set S and $\mathcal{A}_2$ gets as input the set S and oracle access to the query algorithm (initialized with M) and perform adaptive queries. The definition of Bloom filter failure may differ depending on various definitions of robustness, resulting in distinct objectives for $\mathcal{A}_2$ in each scenario. The adversary's runtime and her probability of causing the Bloom filter to fail are considered functions of the security parameter λ. It holds that $n = poly(\lambda)$ (otherwise if $n = superpoly(\lambda)$ the adversary would not even be able to access the entire set).

Definition 5 (The Adaptive Game $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$ (Def 2.2 in [NO22]))**.**

1. The adversary $\mathcal{A}_1$ is given input $1^{\lambda + n \log u}$ (where $u = |U|$) and outputs a set $S \subset U$ of size n.

2. B_1 is given input $(1^{\lambda + \log u}, S)$ and builds a representation M.

[1] If the false positive rate is exceedingly low, then any polynomial-time adversary has a negligible chance of identifying a false positive. Consequently, we can convert any adaptive adversary into a non-adaptive one since they already know the responses. A similar argument is presented in [BLV19] as Lemma 4. When ε is close to 1, a similar reasoning applies.

3. The adversary A_2 is given input $(1^{\lambda+\log u}, S)$ and oracle access to $\mathbf{B}_2(M,\cdot)$ and performs at most t adaptive queries $x_1,\ldots,x_t$ to $\mathbf{B}_2(M,\cdot)$.

We assume wlog that $x_i \notin S$ for all $i \in [t]$, since Bloom filters admit false positives, but not false negatives, and also, $\mathcal{A}_2$ is given as input the set S. Therefore, a queried element can be either a false positive or a true negative.

From now on, we will refer to the adversary as $\mathcal{A}$, without distinguishing between $\mathcal{A}_1$ and $\mathcal{A}_2$.

We consider two types of adversaries: efficient adversaries, which run in polynomial time, and computationally unbounded adversaries, which are only constrained by the number of queries they can make. The parameter t indicates the number of queries that the adversary performs.

The exact definition of security depends on the rules of winning and losing such a game. For a given type of game, we want a Bloom filter that is resilient under its rules; see below the definitions of Always Bet (Section 3.2.1), Bet-or-Pass (Section 3.2.2) and Monotone (Section 3.2.3).

3.2 Robustness Definitions

We aim to define the desirable behavior of Bloom filters with a non-negligible rate of false positives (denoted by ε). We can compare the responses of a Bloom filter to a series of independent biased coin tosses (with probability ε to 1). When considering the adaptive case, our *wishful thinking* is that a Bloom filter still behaves like a truly unpredictable biased coin. Naor and Oved [NO22] encapsulated this idea by proposing various robustness definitions that capture different aspects of robustness. In this work, we examine the relations between some of the definitions they presented.

In the following definitions, we start with an (n,ε)-Bloom filter (under the static definition) and aim to strengthen it to achieve the desired behavior against an adaptive adversary. The definitions we provide refer to the case of unbounded adversaries; extending them to the case of polynomial-time adversaries is straightforward.

3.2.1 The Always-Bet (AB) Test

This test was presented by Naor and Yogev [NY15]: the adversary participates in a game $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$ and then outputs an element x^* that was never queried before. The adversary wins if the probability that x^* is a false positive is high.

The AB Test $\mathsf{ABTest}_{\mathcal{A},t}(\lambda)$:

1. $\mathcal{A}$ participates in the $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$ (Def 5). Let S be the set that the adversary chose and $\{x_1,\ldots,x_t\}$ be the queries that the adversary performed during the game.

2. $\mathcal{A}$ outputs $x^* \notin S \cup \{x_1,\ldots,x_t\}$.

3. The result of the test is 1 if $\mathbf{B}_2(M,x^*)=1$, and 0 otherwise. If $\mathsf{ABTest}_{\mathcal{A},t}(\lambda)=1$, we say that $\mathcal{A}$ made the Bloom filter fail.

Definition 6 (Always Bet (AB) Test (Definition 3.1 in [NO22]))**.** Let $\mathbf{B}=(\mathbf{B}_1,\mathbf{B}_2)$ be an (n,ε)-Bloom filter. We say that $\mathbf{B}$ is (n,t,ε)-Always-Bet (AB) test resilient if for any probabilistic adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$ there exists a negligible function negl such that:

$$\Pr[\mathsf{ABTest}_{\mathcal{A},t}(\lambda)=1] \le \varepsilon + \mathsf{negl}(\lambda)$$

where the probabilities are taken over the internal randomness of $\mathbf{B}$ and $\mathcal{A}$.

An alternative and equivalent way to phrase the AB test is to consider the profit of $\mathcal{A}$ as defined below as one would have for a biased coin with probability ε of being 1 and require that the expected value of the profit be only negligibly larger than 0.

$$C_{\mathcal{A}} = \begin{cases} \frac{1}{\varepsilon}, & \text{if } x^* \text{ is a false positive,} \\ -\frac{1}{1-\varepsilon}, & \text{if } x^* \text{ is } \textbf{not} \text{ a false positive.} \end{cases}$$

3.2.2 The Bet-or-Pass (BP) Test

This test, presented by Naor and Oved [NO22], extends the AB-test (Def 3.2.1): we give the adversary the option to pass; the adversary does not have to output a challenge element x^* (as opposed to the AB-test in which the adversary must output an element). The adversary participates in an $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$ and then outputs (x^*, b) where $x^* \notin S \cup \{x_1, \ldots, x_t\}$ is the challenge and b indicates whether she wants to bet ($b = 1$) or pass ($b = 0$). If she chooses to pass x^* is ignored. To measure the success of the adversary we define her *profit*: if the element she outputs is a false positive she gets rewarded, while she is penalized if she outputs a true negative. If she chooses to pass, she does not gain or lose any value. The expected profit defines the robustness, when we want the expected profit to be 0.

The BP Test $\mathsf{BPTest}_{\mathcal{A},t}(\lambda, \varepsilon)$:

1. $\mathcal{A}$ participate in the $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$ (Def 5). Let S be the set that the adversary chose and $\{x_1, \ldots, x_t\}$ be the queries that the adversary performed during the game.
2. $\mathcal{A}$ outputs (b, x^*) where $x^* \notin S \cup \{x_1, .., x_t\}$.
3. $\mathcal{A}$'s profit $C_{\mathcal{A}}$ is defined as:

$$C_{\mathcal{A}} = \begin{cases} \frac{1}{\varepsilon}, & \text{if } x^* \text{ is a false positive and } b = 1, \\ -\frac{1}{1-\varepsilon}, & \text{if } x^* \text{ is } \textbf{not} \text{ a false positive and } b = 1, \\ 0, & \text{if } b = 0. \end{cases}$$

Definition 7 (Bet-or-Pass (BP) Test (Def 3.2 in [NO22]))**.** Let $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2)$ be an (n, ε)-Bloom filter. We say that $\mathbf{B}$ is (n, t, ε)-Bet-or-Pass (BP) test resilient if for any probabilistic adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ there exists a negligible function negl such that:

$$\mathbb{E}[C_{\mathcal{A}}] \leq \mathsf{negl}(\lambda)$$

when the expectation is taken over the internal randomness of $\mathbf{B}$ and $\mathcal{A}$.

3.2.3 Monotone Tests

To formalize the simile that the Bloom filter behaves as a truly unpredictable biased coin, we consider monotone tests. In these tests, we are not concerned with specific output elements. Instead, we are interested in how the Bloom filter responds to a series of adaptive queries from an adversary. In the Monotone test, the adversary first participates in the $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$, we then consider the Boolean vector $Y = (Y_1, Y_2, \ldots, Y_t)$ consisting of the sequence of responses to the queries that the adversary performed during the game (whether they are false positive or not). We want that for any Monotone test (or distinguisher) D, the probability that D returns '1', when given such a Boolean vector, will be at most the probability that D returns '1' when given an independent biased sequence of the same length and with probability ε of being 1.

We restrict the distinguisher to be a monotone function of its input, since there may be elements where the data structure is known not to err (and we consider it a good property rather than being problematic). We consider monotone distinguishers, which are distinguishers of the form D_f, where $f\colon \{0,1\}^t \to \{0,1\}$ is a monotone function, as defined in Definition 2, and D_f an algorithm computing f. If the requirements we described above hold to any distinguisher of this form, we say that the Bloom filter is Monotone test resilient.

Definition 8 (Monotone test resilient)**.** Let $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2)$ be an (n, ε)-Bloom filter. We say that $\mathbf{B}$ is (n, t, ε)-Monotone test resilient if for every monotone algorithm (distinguisher) $D : \{0,1\}^t \to \{0,1\}$ and every probabilistic adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ participating in the $\mathsf{AdaptiveGame}_{\mathcal{A},t}(\lambda)$ there exists a negligible function negl such that:

$$\Pr_{Y \sim G_{\mathcal{A}}}[D(Y) = 1] - \Pr_{Y \sim \mathrm{Ber}_{\varepsilon,t}}[D(Y) = 1] \leq \mathsf{negl}(\lambda)$$

where $G_{\mathcal{A}}$ is the distribution of the Bloom filter outcomes on the t queries issued by $\mathcal{A}$ and $\mathrm{Ber}_{\varepsilon,t}$ is as defined in Definition 1.

It is natural to compare the notion of Monotone test resilience with that of cryptographic pseudorandomness (see Goldreich, Chapter 3 [Gol01]). Pseudorandomness refers to the property of sequences or objects that appear random to any efficient observer, even though they are generated by a deterministic process using only a short seed. This idea of *computational indistinguishability*, shares some similarities with the definition of Monotone test resilience: Given a distribution over t-bit strings Z, we say that Z is pseudorandom if no polynomial-time algorithm can distinguish between a string sampled from the distribution Z and a uniform t-bit string. In contrast to monotone tests, the distinguisher in the context of pseudorandomness is not required to be monotone, and we consider the absolute value of the difference in probabilities. Additionally, in the Monotone test, we have an asymmetry between 0's and 1's.

3.3 Resilience of Bloom Filters Against Different Adversaries

How does the computational power of the adaptive adversary affect the resiliency of the scheme? Naor and Yogev [NY15] proved the impossibility of (n, t, ε)-AB test resilient Bloom filters with memory m when $t \geq \Omega(m/\varepsilon_0^2)$ where ε_o is the minimal error (the Bloom filter's inherent false positive rate given the parameters, see Definition 4) and $\varepsilon < 1$, if the adversary is computationally powerful. More specifically, if the Bloom filter is computationally efficient (i.e. polynomial time), the power the adversary needs is inverting one-way functions. In other words, one must either limit t as a function of m or limit the computational power of the adversary.

On the other hand, they presented an efficient construction that uses pseudo-random functions to obtain an AB-test resilient Bloom filter against any polynomial number of queries when the adversary is computationally bounded. When the adversary is computationally unbounded they gave a construction for an (n, t, ε)-AB test resilient Bloom filter with $m \in O(n \log(1/\varepsilon) + t)$ memory (see the construction in Section 4.1.1). Thus, when t is not known in advance or large compared to m, AB-resilient Bloom filters are possible if and only if the adversary is computationally bounded and pseudo-random functions exist (which are existentially equivalent to one-way functions).

Naor and Oved [NO22] (Section 5 there) showed that this equivalence result still holds given a Bloom filter that is BP-test resilient.

As a result, we focus on two types of adversaries:

- **Computationally unbounded adversaries** limited to t queries, where t is fixed and known in advance.

- **Efficient adversaries** that run in polynomial time, where t is bounded only by the adversary's running time.

A Bloom filter that is resilient to t queries (without restrictions on the adversary's computational power), is called *t-resilient*. And a Bloom filter that is resilient for any polynomial number of queries is called *strongly resilient*; in this case the adversary must be efficient.

4 Relationships: Monotone Test Resilience and its Implications

We now examine the relationships between the notions introduced in Section 3. Naor and Oved, in their work [NO22], demonstrated some of the connections between different definitions, which are highlighted in blue in Figure 1. In this work we complete the picture by investigating the Monotone test resilience implications for AB and BP test resilience, our results marked with black in Figure 1.

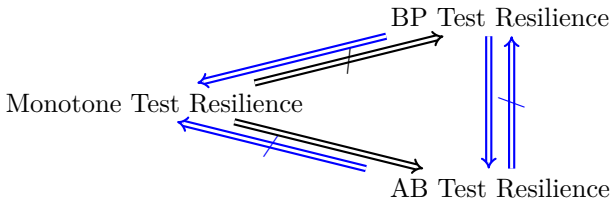

Figure 1: The relationships between the notions; the new results are in black.

Theorem 2 below, showing the bottom arrow, i.e. that monotone resilience implies AB-Test resilience is quite straightforward. We devote most of the space (Section 4.1) to showing the top result, that monotone resilience does not imply BP-test resilience.

Theorem 2. *Let $0 < \varepsilon < 1$ and $n \in \mathbb{N}$. Let $\mathbf{B}$ be an (n, ε)-strongly Monotone test resilient Bloom filter. Then $\mathbf{B}$ is (n, ε)-strongly AB test resilient.*

Proof. Suppose, toward contradiction, that $\mathbf{B}$ is not (n, ε)-strongly AB test resilient Bloom filter. So, there exists a probabilistic polynomial-time adversary $\mathcal{A}$ in the $\mathsf{ABTest}_{\mathcal{A},t}(\lambda)$ where $t \leq poly(n, \lambda)$ and a polynomial $p(\cdot)$ such that for infinitely many λ's:

$$\Pr[x^* \text{ is False Positive}] \geq \varepsilon + \frac{1}{p(\lambda)}$$

We now define a monotone polynomial-time test D_{t+1} that distinguishes $G_{\mathcal{A}}$, the vector s.t. the ith entry is the answer to the ith query of $\mathcal{A}$ (including the answer to x^*), from $\mathrm{Ber}_{\varepsilon,t+1}$ the biased independent sequence of length $t+1$ with probability ε of being 1, as in Definition 1.

$$D_{t+1} = \begin{cases} 1, & \text{if the } (t+1)\text{th entry in the sequence is 1} \\ 0, & \text{else} \end{cases}$$

Clearly D_{t+1} is monotone. We show that $\mathbf{B}$ fails the test D_{t+1}, meaning $\mathbf{B}$ is not (n, ε)-strongly Monotone test resilient Bloom filter:

$$\Pr_{Y \sim G_{\mathcal{A}}}[D_{t+1}(Y) = 1] - \Pr_{Y \sim \mathrm{Ber}_{\varepsilon,t+1}}[D_{t+1}(Y) = 1] = \Pr[x^* \text{ is FP}] - \varepsilon \geq \varepsilon + \frac{1}{p(\lambda)} - \varepsilon = \frac{1}{p(\lambda)}$$

This contradicts our assumption.

□

Remark 1. Our proof holds for unbounded adversaries as well, if t is fixed and known in advance.

4.1 Resilience to Monotone tests does *not* imply one to BP tests

As promised, we show that resilience to the Monotone test does *not* imply resilience to the BP test. We will prove this by constructing a Bloom filter that has two possible modes or states: 'bad' and 'good', and show that the 'good' and 'bad' scenarios cannot be distinguished in a monotone manner. In the 'good' state the false positive (FP) rate is slightly below ε, and an adversary has no specific advantage in finding false positives. However, with a small but non-negligible probability, the Bloom filter might be in the 'bad' state where the FP rate for most queries is even lower, but on a 'special' element $\widehat{x}$ the Bloom filter always returns 'Yes', allowing an adversary to exploit this knowledge for betting. A (non-monotone) adversary can identify whether the current state is 'good' or 'bad' by evaluating the FP rate on a randomly chosen query set (that should not include the special element); if the Bloom filter is in the 'bad' state, the adversary can confidently bet on the special element. This shows that the Bloom filter is not resilient to the BP test. On the other hand, we show that no *monotone* test can exploit this weakness.

To actually construct the Bloom filter in this example we need to start with one where we have a good understanding of the false positive rate on random queries. That is, we should know during construction the value of this parameter with good accuracy (it should not be much smaller or larger than the given value).

We note that the BP adversary we suggest does not need to be adaptive in selecting queries to determine the state; instead, she can choose queries randomly from $U \setminus S \cup \{\widehat{x}\}$. The adaptiveness required is only in deciding whether to place a bet or not, based on the observed responses. In more detail:

Theorem 3. *Let $0 < \varepsilon < 1/2$ and $n \in \mathbb{N}$, then for any $0 < \delta < 1/2$ and for large enough u:*

1. *Assuming the existence of one-way functions, there exists a non-trivial Bloom filter* $\mathbf{B}$ *that is (n, ε)-strongly Monotone test resilient and is* not *an (n, δ)-strongly BP test resilient.*
2. *If t is fixed and satisfies $t > \frac{32 \cdot \ln \frac{2}{\varepsilon}}{\varepsilon}$ and $u > t^2$, then there exists a non-trivial Bloom filter* $\mathbf{B}$ *that is (n, t, ε)-Monotone test resilient and is not an (n, t, δ)-BP test resilient.*

Proof. First, assume that t is fixed and known in advance. Let $0 < \varepsilon < 1/2$. We will show a construction of a Bloom filter $\mathbf{B}$ that is (n, t, ε)-Monotone test resilient and is *not* (n, t, δ)-BP test resilient for any $0 < \delta < 1/2$.

For any $n, t \in \mathbb{N}$ and $0 < \varepsilon < 1/2$ Naor and Yogev [NY15] constructed a non-trivial Bloom filter (as defined in definition 4) that is (n, t, ε)-AB test resilient against an *unbounded adversary given number t of queries*. Naor and Oved [NO22] pointed out that under the same restriction (the adversary being limited to t queries), this construction is actually (n, t, ε)-BP test resilient (Section 5.1 there).

As mentioned before, for the counterexample, we need a Bloom filter whose false positive rate is well understood. The construction by Naor and Yogev is close to fulfilling this requirement, as it allows us to derive good bounds on the false positive rate. We use this construction with some modifications that allow us to compute the exact false positive rate. Next We provide a brief description of this construction (more details can be found in Section 6 of [NY15]) and the modifications needed.

4.1.1 Overview of the (n, t, ε)-BP test resilient Bloom filter construction

Naor and Yogev's construction is based on two main ingredients: Cuckoo Hashing dictionary [PR04], this structure consists of two tables T_1 and T_2 that maintain the elements and two hash functions h_1 and h_2. Each element is stored either at $T_1[h_1(x)]$ or $T_2[h_2(x)]$. The second ingredient is a family G of hash functions satisfying that on any set of k inputs, it behaves like a truly random function with high probability [Pag08, DW03] (even when the set of queries is chosen adaptively [BHKN19]). We use an exact k-wise independent function for the family G and take k to be $t + n + 1$ (see below). Each $g \in G$ maps a pair (x, j) where $x \in U$ and $j \in \{1, 2\}$ to ℓ bit strings.

The idea is to store the set S in a Cuckoo Hashing dictionary, where each element x is stored in either $T_1[h_1(x)]$ or $T_2[h_2(x)]$. To save space, instead of storing x, we store the evaluation of a function g at point x, where g is randomly chosen from the family G described above. Specifically, if x is stored in T_1, we store the value $g(x, 1)$, otherwise we store the value $g(x, 2)$. In the empty cells of the tables, we put randomly chosen ℓ-bit strings. We choose $k = t + n + 1$ and $\ell = \log \frac{2}{\varepsilon}$.

Given a query x, we compare $g(x, 1)$ with $T_1[h_1(x)]$ and $g(x, 2)$ with $T_2[h_2(x)]$; if either of them is equal a 'yes' is returned.

Modifications and Simplifications. For our counterexample, we made several simplifications and modifications to the original construction [NY15] to fit our purpose (or proof) and obtain the property that there is some $\varepsilon' \leq \varepsilon$ where for any query $x \notin S$ and any history of at most t queries the probability of false positive is exactly ε'. Here is the summary of the changes:

Exact Independence: We use an exact k-wise independent function instead of an almost k-wise independent. Additionally, we drop the optimization of making G composed from one-bit functions. In the lookup phase, we simplify the process by avoiding the bit-by-bit comparison used in Naor and Yogev's construction.

Modifying the parameters: The parameters k and ℓ are modified. In the original construction, they were $k = 2t/\log \frac{1}{\varepsilon}$ and $\ell = 4 \log \frac{1}{\varepsilon}$.

Random values in empty bins: We put a random value in the empty locations of the tables, instead of the $\perp$ value that was used in [NY15], in order to have an exact evaluation of the probability of a false positive response.

Different values for T_1 and T_2: We want to avoid a potential bias in the usual implementation arising from knowing that two values in the table are the same, which slightly reduces the probability of collision. Hence, we modify the original construction and store different applications of the function g in the first table and the second table: $g(x, 1)$ and $g(x, 2)$. Therefore, knowledge of the contents of the tables does not change the conditional probability of false positive.

In the modified construction, for every $x \in S$ we store either $g(x, 1)$ in $T_1[h_1(x)]$ or $g(x, 2)$ in $T_2[h_2(x)]$. Hence, when querying $x \in S$ we have,

$$\Pr\left[g(x, 1) = T_1[h_1(x)] \text{ OR } g(x, 2) = T_2[h_2(x)]\right] = 1$$

ensuring that there are no false negatives. Moreover, as we prove in Lemma 1, we can compute the exact probability of a false positive.

Call a Bloom filter constructed as above (n, t, ε)-**CHBF**. What Naor and Oved showed is that this construction is indeed (n, t, ε)-BP test resilient. Our point is that the probability of a false positive is not much lower than ε:

Lemma 1. *Let $\mathbf{B}_{NY}$ be an (n,t,ε)-CHBF and $\mathcal{A}$ an adversary that performs t adaptive distinct queries $x_1,\ldots,x_t \in U \setminus S$. Let $\varepsilon' = \varepsilon - \frac{\varepsilon^2}{4}$. Then for* any *query x_i given the history of the results of queries $x_1,\ldots x_{i-1}$ the probability that x_i is a false positive is exactly ε' and*

$$(Y_1,\ldots,Y_t) \sim Ber_{\varepsilon',t}$$

where $Y_1,\ldots,Y_t$ are the (Boolean) responses of $\mathbf{B}_{NY}$ to the queries $x_1,\ldots,x_t$.

Proof. For any query x_i the probability that x_i is a false positive equals the probability that $g(x_i,1) = T_1[h_1(x_i)]$ or $g(x_i,2) = T_2[h_2(x_i)]$. Since g is k-wise independent and $t+n \le k$, even if all the values in the tables that have ever been used are known to the adversary, the value of $g(x_i,j)$ remains uniformly distributed over its range. Thus,

$$\begin{aligned}\Pr[x_i \text{ is FP}] =& \Pr\left[g(x_i,1) = T_1[h_1(x_i)] \text{ OR } g(x_i,2) = T_2[h_2(x_i)]\right]\\ =& 2\cdot 2^{-\ell} - (2^{-\ell})^2\\ =& 2\cdot 2^{-(\log\frac{2}{\varepsilon})} - \left(2^{-(\log\frac{2}{\varepsilon})}\right)^2\\ =& \varepsilon - \frac{\varepsilon^2}{4}\end{aligned}$$

where the first equality is derived from the inclusion-exclusion principle.

Since g is k-wise independent and $t+n<k$, then $Y_1,\ldots,Y_t$ are independent and for every i it holds that Y_i is distributed as Bernoulli with probability $\varepsilon - \frac{\varepsilon^2}{4}$ of being 1. Hence,

$$(Y_1,\ldots,Y_t) \sim Ber_{\varepsilon',t}$$

□

4.1.2 The Counter-Example:

For convenience, we assume wlog that ε is a power of 2 (otherwise, we can take the closest smaller power of 2). We use the CHBF construction above to get two Bloom Filters: (i) $\mathbf{B}'$ that is $(n,t,\frac{\varepsilon}{16})$-CHBF and (ii) $\mathbf{B}''$ that is $(n,t,\frac{\varepsilon}{2})$-CHBF.

For a fixed known element $\widehat{x} \in U \setminus S$ we modify $\mathbf{B}'$ so that when queried on $\widehat{x}$ it will answer 1 w.p. 1 (achieved by simply inserting $\widehat{x}$ into $\mathbf{B}'$). Using $\mathbf{B}'$ and $\mathbf{B}''$ we construct,

$$\mathbf{B}(S,\cdot) \equiv \begin{cases}\mathbf{B}'(S,\cdot), & \text{w.p. } \frac{\varepsilon}{2}\\ \mathbf{B}''(S,\cdot), & \text{w.p. } 1-\frac{\varepsilon}{2}\end{cases}$$

That is, in the setup phase $\mathbf{B}$ flips a coin with probability $\frac{\varepsilon}{2}$ of being 1, which determines whether it will behave like $\mathbf{B}'$ or $\mathbf{B}''$. During the query phase, $\mathbf{B}$ responds to each query in the same way as the Bloom filter selected in the setup phase. We denote by $\mathbf{B} \equiv \mathbf{B}'$ the case where $\mathbf{B}$ answers as $\mathbf{B}'$ (resp. $\mathbf{B} \equiv \mathbf{B}''$).

Lemma 2. $\mathbf{B}$ *is not (n,t,δ)-BP test resilient for any $0<\delta<1/2$.*

Proof. Let $0<\delta<1/2$. We describe an adversary $\mathcal{A}$ that makes $\mathbf{B}$ fail the (n,t,δ)-BP test: $\mathcal{A}$ queries t random elements in $U \setminus S \cup \{\widehat{x}\}$ to determine whether $\mathbf{B} \equiv \mathbf{B}'$ or $\mathbf{B} \equiv \mathbf{B}''$. If $\mathbf{B} \equiv \mathbf{B}'$, then $\mathcal{A}$ bets on the special element $\widehat{x}$; Otherwise, she passes. In more detail:

Adversary $\mathcal{A}$:

1. Choose a random set $S \subset U \setminus \{\widehat{x}\}$ of size n
2. Query independent random elements $\{x_1,\ldots,x_t\} \subseteq U \setminus S \cup \{\widehat{x}\}$ and get Y_i $1 \le i \le t$
3. If $\sum_{i=1}^{t} Y_i \le \frac{\varepsilon}{8}\cdot t$, then output $(b=1),\widehat{x}$

4. Otherwise, output $(b = 0), \widehat{x}$

Each response Y_i the adversary gets is a random variable indicating whether the ith query is a false positive. Since the adversary $\mathcal{A}$ chooses independent random elements to query and the universe U is large, $u > t^2$, we can assume that w.h.p. she does not query the same element twice (by the Birthday paradox, Chapter 5 in [MU05]). Hence, by Lemma 1, when $\mathbf{B} \equiv \mathbf{B}''$ it holds that $Y_1, \ldots, Y_t \sim Ber_{\left(\frac{\varepsilon}{2} - \frac{\varepsilon^2}{16}\right), t}$. Furthermore, since $\widehat{x} \notin \{x_1, \ldots, x_t\}$, the same analysis as in the proof of Lemma 1 implies that when $\mathbf{B} \equiv \mathbf{B}'$ it holds that $Y_1, \ldots, Y_t \sim Ber_{\left(\frac{\varepsilon}{16} - \frac{\varepsilon^2}{1024}\right), t}$. Note that this gap between the probabilities allows the adversary to know whether we are in the case $\mathbf{B} \equiv \mathbf{B}'$ or $\mathbf{B} \equiv \mathbf{B}''$ w.h.p. Using the Chernoff bound specified in Theorem 1, we get that the probability to bet, i.e that $b = 1$, is at least $\left(1 - e^{(\varepsilon/32)\cdot t}\right)$ when $\mathbf{B} \equiv \mathbf{B}'$, and at most $e^{(\varepsilon/32)\cdot t}$ when $\mathbf{B} \equiv \mathbf{B}''$. Recall that $C_{\mathcal{A}}$ denotes $\mathcal{A}$'s profit in the BP-test. Then,

$$
\begin{aligned}
\mathbb{E}[C_{\mathcal{A}}] &= \mathbb{E}[C_{\mathcal{A}} | \mathbf{B} \equiv \mathbf{B}'] \cdot \Pr[\mathbf{B} \equiv \mathbf{B}'] + \mathbb{E}[C_{\mathcal{A}} | \mathbf{B} \equiv \mathbf{B}''] \cdot \Pr[\mathbf{B} \equiv \mathbf{B}''] \\
&= \frac{1}{\delta} \cdot \Pr[\mathbf{B} \equiv \mathbf{B}'] \cdot \Pr[\text{b=1 } | \mathbf{B} \equiv \mathbf{B}'] \cdot \Pr[x^* \text{ is FP} | \text{b=1 And } \mathbf{B} \equiv \mathbf{B}'] \\
&- \frac{1}{1-\delta} \cdot \Pr[\mathbf{B} \equiv \mathbf{B}'] \cdot \Pr[\text{b=1 } | \mathbf{B} \equiv \mathbf{B}'] \cdot \underbrace{\Pr[x^* \text{ is not FP} | \text{b=1 And } \mathbf{B} \equiv \mathbf{B}']}_{=0} \\
&+ \frac{1}{\delta} \cdot \Pr[\mathbf{B} \equiv \mathbf{B}''] \cdot \Pr[\text{b=1 } | \mathbf{B} \equiv \mathbf{B}''] \cdot \underbrace{\Pr[x^* \text{ is FP} | \text{b=1 And } \mathbf{B} \equiv \mathbf{B}'']}_{\geq 0} \\
&- \frac{1}{1-\delta} \cdot \Pr[\mathbf{B} \equiv \mathbf{B}''] \cdot \Pr[\text{b=1 } | \mathbf{B} \equiv \mathbf{B}''] \cdot \underbrace{\Pr[x^* \text{ is not FP} | \text{b=1 And } \mathbf{B} \equiv \mathbf{B}'']}_{\leq 1} \\
&\geq \frac{1}{\delta} \cdot \frac{\varepsilon}{2} \cdot \Pr[\text{b=1 } | \mathbf{B} \equiv \mathbf{B}'] - \frac{1}{1-\delta} \cdot \left(1 - \frac{\varepsilon}{2}\right) \cdot \Pr[\text{b=1 } | \mathbf{B} \equiv \mathbf{B}''] \\
&\geq \frac{1}{\delta} \cdot \frac{\varepsilon}{2} \cdot \left(1 - e^{(\varepsilon/32)\cdot t}\right) - \frac{1}{1-\delta} \cdot \left(1 - \frac{\varepsilon}{2}\right) \cdot e^{(\varepsilon/32)\cdot t} \\
&\geq \frac{1}{1-\delta} \cdot \left[\frac{\varepsilon}{2} - e^{(\varepsilon/32)\cdot t}\right]
\end{aligned}
$$

So for sufficiently large t, $\mathbb{E}[C_{\mathcal{A}}]$ is non-negligible, and $\mathcal{A}$ wins the game. □

Lemma 3. **B** *is* (n, t, ε)*-Monotone test resilient.*

Proof. We will show that for any *monotone* distinguisher D, every adversary $\mathcal{A}$ that makes at most t queries it holds that:

$$\Pr_{Y \sim G_{\mathcal{A}}}[D(Y) = 1] - \Pr_{Y \sim \text{Ber}_{\varepsilon,t}}[D(Y) = 1] \leq \mathsf{negl}(\lambda)$$

When $G_{\mathcal{A}}$ is the vector s.t. the ith entry is the answer to the ith query of $\mathcal{A}$ and $\text{Ber}_{\varepsilon,t}$ the biased independent sequence of length t with probability ε of being 1, as in Definition 1.

Let D be a monotone distinguisher and $\mathcal{A}$ an adversary that makes at most t queries. We assume that $\mathcal{A}$ is deterministic since we can fix the best choice of randomness (the one maximizing the probability of success). We can further assume wlog that $\mathcal{A}$ queries the point $\widehat{x}$, since if $\mathcal{A}$ does not query it within the first t queries we can add an additional query such that $\mathcal{A}$ queries $\widehat{x}$ in the $(t+1)$th query.

Let $\widehat{k}$ be the index of the query in which $\mathcal{A}$ asked about $\widehat{x}$ and let

- $p_1 = \Pr_{Y \sim G_{\mathcal{A}}}[D(Y) = 1 | \mathbf{B} \equiv \mathbf{B}']$

- $q_1 = \Pr_{Y\sim \mathrm{Ber}_{\varepsilon,t}}[D(Y)=1|Y_{\widehat{k}}=1]$
- $q_0 = \Pr_{Y\sim \mathrm{Ber}_{\varepsilon,t}}[D(Y)=1|Y_{\widehat{k}}=0]$.

Recall from Section 2 that for monotone D it holds that:

$$\forall 0 \le \alpha \le \varepsilon \qquad \Pr_{Y\sim \mathrm{Ber}_{\alpha,t}}[D(Y)=1] \le \Pr_{Y\sim \mathrm{Ber}_{\epsilon,t}}[D(Y)=1].$$

Due to the symmetry in the Bernoulli case $\mathrm{Ber}_{\varepsilon,t}$, we can assume wlog that $\widehat{k}=t$ When the $\widehat{k}$th entry is fixed, a Monotone test D that looks only on the other $t-1$ bits remains monotone. We have:

$$\Pr_{Y\sim \mathrm{Ber}_{\varepsilon/2,t}}[D(Y)=1|Y_{\widehat{k}}=1] \le q_1$$

and

$$\Pr_{Y\sim \mathrm{Ber}_{\varepsilon/2,t}}[D(Y)=1|Y_{\widehat{k}}=0] \le q_0.$$

If $\mathbf{B}''$ is $(n,t,\frac{\varepsilon}{2})$-**BP** test resilient, then it is also $(n,t,\frac{\varepsilon}{2})$-**Monotone** test resilient (Section 4.1.1 in [NO22]). It holds that:

$$\Pr_{Y\sim G_{\mathcal{A}}}[D(Y)=1|\mathbf{B}\equiv\mathbf{B}''] - \Pr_{Y\sim \mathrm{Ber}_{\varepsilon/2,t}}[D(Y)=1] \le \mathsf{negl}(\lambda)$$

Thus,

$$\Pr_{Y\sim G_{\mathcal{A}}}[D(Y)=1|\mathbf{B}\equiv\mathbf{B}''] \le q_1\cdot\frac{\varepsilon}{2} + q_0\cdot\left(1-\frac{\varepsilon}{2}\right) + \mathsf{negl}(\lambda)$$

By the law of total probability, we have that:

$$\begin{aligned}
\Pr_{Y\sim G_{\mathcal{A}}}[D(Y)=1] - \Pr_{Y\sim \mathrm{Ber}_{\varepsilon,t}}[D(Y)=1] &\le p_1\cdot\frac{\varepsilon}{2} + \left[q_1\cdot\frac{\varepsilon}{2} + q_0\cdot\left(1-\frac{\varepsilon}{2}\right) + \mathsf{negl}(\lambda)\right]\cdot\left(1-\frac{\varepsilon}{2}\right) \\
&\quad - q_1\cdot\varepsilon - q_0\cdot(1-\varepsilon) \\
&= p_1\cdot\frac{\varepsilon}{2} + \left(-\frac{\varepsilon}{2}-\frac{\varepsilon^2}{4}\right)\cdot q_1 + \frac{\varepsilon^2}{4}\cdot q_0 + \mathsf{negl}(\lambda) \\
&\le p_1\cdot\frac{\varepsilon}{2} + \left(-\frac{\varepsilon}{2}-\frac{\varepsilon^2}{4}\right)\cdot q_1 + \frac{\varepsilon^2}{4}\cdot q_1 + \mathsf{negl}(\lambda) \qquad (*) \\
&= p_1\cdot\frac{\varepsilon}{2} - q_1\cdot\frac{\varepsilon}{2} + \mathsf{negl}(\lambda).
\end{aligned}$$

Where $(*)$ is derived from the fact that $q_0 \le q_1$ since D is monotone. When the $\widehat{k}$th entry is fixed, we consider only the results of queries that differ from $\widehat{x}$. In this case, we can bound the probability of false positive by the same analysis as in the proof of Lemma 1. Hence, the probability of each query being a false positive is bounded by $\frac{\varepsilon}{16}$ and $p_1 - q_1 \le \mathsf{negl}(\lambda)$, as desired.

□

We proved that $\mathbf{B}$ is (n,t,ε)-Monotone test resilient but not (n,t,ε)-BP test resilient as desired.

When t is unknown and unbounded, we must resort to computational assumptions, as proved in [NY15]. We need to use Naor and Oved's construction of a BP strongly resilient Bloom filter (which in turn is the one in [NY15]), based on the existence of

one-way functions; roughly speaking, the k-wise independent function is replaced with a pseudorandom function (PRF). The construction of $\mathbf{B}$ is the same except that $\mathbf{B}'$, $\mathbf{B}''$ are BP strongly resilient. The same proof holds because, for any polynomial number of queries t, the PRF is indistinguishable from a $2t$-wise independent hash function. In particular, the probability shown in Lemma 1 can also be proven when the function is a PRF instead of a $2t$-wise independent function for a polynomial number of queries. Additionally, the same adversary as we describe in the proof of Lemma 2 can be used to demonstrate that this construction is not strongly BP-test resilient, as the number of queries required to distinguish between the cases $\mathbf{B} \equiv \mathbf{B}'$ and $\mathbf{B} \equiv \mathbf{B}''$ is polynomial in n. Finally, the proof for Monotone resilience holds for any t, and therefore applies in this case as well.

□

Conclusions and Open Directions

There has been a surge of research regarding the power of adversaries across various areas, including data structures, streaming algorithms, and machine learning (see, for example [CPS19, BJWY22, BY20, KMNS21, ABD+21]). This raises the question of how relevant are the notions of robustness against adversaries and their relationships, as discussed in this paper and in [NO22] for those different settings.

Bishop and Tirmazi [BT24] introduced an adversarial model for a variant of the Bloom filter, known as the 'Learned Bloom filter' (see [KBC+18, Mit18]), and proposed a construction that satisfies it. The Learned Bloom filter can be thought of as a Bloom filter working in collaboration with a Learning Model, where the idea is to use a pre-filter (derived from the model) before the Bloom filter. This approach improves the false positive rate while maintaining zero false negatives. The adversarial model suggested by Bishop and Tirmazi is based on the framework of Naor and Yogev (the AB-test); the question is what is the robustness of the constructions based on Learned Bloom filters w.r.t. the notions discussed in this paper, in particular BP.

Filic et al. [FPUV22] propose security definitions for probabilistic data structures, focusing on both correctness and privacy in a simulation-based framework. They use these definitions to analyze the behavior of Bloom filters and insertion-only Cuckoo filters. In their model, the adversary can query the filter, insert new elements, and observe its current state (but there seems to be an area in memory that is not accessible and where cryptographic keys can be stored). While their definition appears to grant the adversary more power compared to ours, it does not imply BP-test resilience. For instance, a Bloom filter that satisfies their definition most of the time but with non-negligible probability enters a state where it always answers 'Yes' would still satisfy their definition but fails the BP-test.

Repeated queries: Adversarial environments for Bloom filters were further investigated by Bender et al. [BFCG+18]. Their adversarial model is similar to the one of Naor and Yogev (the AB-test), but unlike all the definitions considered in this paper (in AB,BP and Monotone settings), they allow the adversary to repeat any query. Their main concern was that an adversary could achieve a false positive rate close to 1 by repeatedly querying false positives. To handle this, they proposed an *adaptive filter*, a structure that adjusts to false positives. This adaptive filter ensures that the probability of a false positive remains at most ε, even when repeating a previously queried element. In their analysis, they assume that the adversary cannot find a never-queried-before element that yields a false positive with probability greater than ε, even when using results from previous queries. This assumption is achievable through the construction of Naor and Yogev. The question is whether their results are compatible with the findings in this paper and in Naor and Oved, and whether it is possible to construct a robust BP-resilient Bloom filter that withstands repeated queries.

Acknowledgments

We thank the referees for their helpful comments improving the presentation of the paper.

References

[ABD+21] Noga Alon, Omri Ben-Eliezer, Yuval Dagan, Shay Moran, Moni Naor, and Eylon Yogev. Adversarial laws of large numbers and optimal regret in online classification. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 447–455. ACM, 2021. doi:10.1145/3406325.3451041.

[BFCG+18] Michael A. Bender, Martin Farach-Colton, Mayank Goswami, Rob Johnson, Samuel McCauley, and Shikha Singh. Bloom filters, adaptivity, and the dictionary problem. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 182–193, 2018. doi:10.1109/FOCS.2018.00026.

[BHKN19] Itay Berman, Iftach Haitner, Ilan Komargodski, and Moni Naor. Hardness-preserving reductions via cuckoo hashing. *J. Cryptol.*, 32(2):361–392, 2019. URL: https://doi.org/10.1007/s00145-018-9293-0, doi:10.1007/S00145-018-9293-0.

[BJWY22] Omri Ben-Eliezer, Rajesh Jayaram, David P. Woodruff, and Eylon Yogev. A framework for adversarially robust streaming algorithms. *J. ACM*, 69(2):17:1–17:33, 2022. doi:10.1145/3498334.

[Blo70] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, 1970. doi:10.1145/362686.362692.

[BLV19] Elette Boyle, Rio LaVigne, and Vinod Vaikuntanathan. Adversarially robust property-preserving hash functions. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 16:1–16:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. URL: https://doi.org/10.4230/LIPIcs.ITCS.2019.16, doi:10.4230/LIPICS.ITCS.2019.16.

[BM03] Andrei Z. Broder and Michael Mitzenmacher. Survey: Network applications of bloom filters: A survey. *Internet Math.*, 1(4):485–509, 2003. doi:10.1080/15427951.2004.10129096.

[BT24] Allison Bishop and Hayder Tirmazi. Adversary resilient learned Bloom filters. *IACR Cryptol. ePrint Arch.*, page 754, 2024. URL: https://eprint.iacr.org/2024/754.

[BY20] Omri Ben-Eliezer and Eylon Yogev. The adversarial robustness of sampling. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2020, Portland, OR, USA, June 14-19, 2020*, pages 49–62. ACM, 2020. doi:10.1145/3375395.3387643.

[CFG+78] Larry Carter, Robert W. Floyd, John Gill, George Markowsky, and Mark N. Wegman. Exact and approximate membership testers. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 59–65. ACM, 1978. doi:10.1145/800133.804332.

[CPS19] David Clayton, Christopher Patton, and Thomas Shrimpton. Probabilistic data structures in adversarial environments. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 1317–1334, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3319535.3354235.

[DW03] Martin Dietzfelbinger and Philipp Woelfel. Almost random graphs with simple hash functions. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 629–638. ACM, 2003. doi:10.1145/780542.780634.

[FPUV22] Mia Filic, Kenneth G. Paterson, Anupama Unnikrishnan, and Fernando Virdia. Adversarial correctness and privacy for probabilistic data structures. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, page 1037–1050, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3548606.3560621.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001. URL: http://www.wisdom.weizmann.ac.il/%7Eoded/foc-vol1.html, doi:10.1017/CBO9780511546891.

[KBC+18] Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The case for learned index structures. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 489–504. ACM, 2018. doi:10.1145/3183713.3196909.

[KMNS21] Haim Kaplan, Yishay Mansour, Kobbi Nissim, and Uri Stemmer. Separating adaptive streaming from oblivious streaming using the bounded storage model. In *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pages 94–121. Springer, 2021. doi:10.1007/978-3-030-84252-9_4.

[KS06] Gil Kalai and Shmuel Safra. Threshold phenomena and influence: Perspectives from mathematics, computer science, and economics. In Allon G. Percus, Gabriel Istrate, and Cristopher Moore, editors, *Computational Complexity and Statistical Physics*, Santa Fe Institute Studies in the Sciences of Complexity, pages 25–62. Oxford University Press, 2006.

[Mit18] Michael Mitzenmacher. A model for learned Bloom filters and optimizing by sandwiching. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 462–471, 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/0f49c89d1e7298bb9930789c8ed59d48-Abstract.html.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005. doi:10.1017/CBO9780511813603.

[NO22] Moni Naor and Noa Oved. Bet-or-pass: Adversarially robust Bloom filters. In *Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part II*, volume 13748 of *Lecture Notes in Computer Science*, pages 777–808. Springer, 2022. doi:10.1007/978-3-031-22365-5_27.

[NPB21] Sabuzima Nayak, Ripon Patgiri, and Angana Borah. A survey on the roles of Bloom filter in implementation of the named data networking. *Computer Networks*, 196:108232, 2021. URL: https://www.sciencedirect.com/science/article/pii/S1389128621002747, doi:10.1016/j.comnet.2021.108232.

[NY15] Moni Naor and Eylon Yogev. Bloom filters in adversarial environments. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 565–584. Springer, 2015. doi:10.1007/978-3-662-48000-7_28.

[Pag08] Rasmus Pagh. Cuckoo hashing. In Ming-Yang Kao, editor, *Encyclopedia of Algorithms - 2008 Edition*. Springer, 2008. doi:10.1007/978-0-387-30162-4_97.

[PR04] Rasmus Pagh and Flemming Friche Rodler. Cuckoo hashing. *J. Algorithms*, 51(2):122–144, 2004. URL: https://doi.org/10.1016/j.jalgor.2003.12.002, doi:10.1016/J.JALGOR.2003.12.002.

[TRL12] Sasu Tarkoma, Christian Esteve Rothenberg, and Eemil Lagerspetz. Theory and practice of bloom filters for distributed systems. *IEEE Commun. Surv. Tutorials*, 14(1):131–155, 2012. doi:10.1109/SURV.2011.031611.00024.