Tight Time-Space Lower Bounds for Finding Multiple Collision Pairs and Their Applications

Itai Dinur

Department of Computer Science, Ben-Gurion University, Israel

Abstract. We consider a *collision search problem* (CSP), where given a parameter C, the goal is to find C collision pairs in a random function $f : [N] \rightarrow [N]$ (where $[N] = \{0, 1, \ldots, N-1\}$) using S bits of memory. Algorithms for CSP have numerous cryptanalytic applications such as space-efficient attacks on double and triple encryption. The best known algorithm for CSP is *parallel collision search* (PCS) published by van Oorschot and Wiener, which achieves the time-space tradeoff $T^2 \cdot S = \tilde{O}(C^2 \cdot N)$ for $S = \tilde{O}(C)$.

In this paper, we prove that any algorithm for CSP satisfies $T^2 \cdot S = \tilde{\Omega}(C^2 \cdot N)$ for $S = \tilde{O}(C)$, hence the best known time-space tradeoff is optimal (up to poly-logarithmic factors in N). On the other hand, we give strong evidence that proving similar unconditional time-space tradeoff lower bounds on CSP applications (such as breaking double and triple encryption) may be very difficult, and would imply a breakthrough in complexity theory. Hence, we propose a new restricted model of computation and prove that under this model, the best known time-space tradeoff attack on double encryption is optimal.

Keywords: Collision search problem, time-space tradeoff, R-way branching program, provable security, cryptanalysis, parallel collision search, double encryption.

1 Introduction

A time-space tradeoff for a problem is a curve that quantifies the difficulty of solving it in terms of the required time complexity T and space complexity S (and perhaps additional problem-specific parameters). Such tradeoffs play a significant role in algorithmic research, as they provide a more realistic estimate of how difficult it is to solve a problem by considering the available space, as opposed to analysis that only considers the available computation power.

In this work we consider time-space tradeoffs for the *collision search problem* (CSP), where given a parameter C and oracle access to a random function $f : [N] \rightarrow [N]$, the goal is to find C distinct unordered colliding pairs $(i_1, i_2) \in [N]^2$ in f (i.e., $i_1 \neq i_2$, but $f(i_1) = f(i_2)$) using S bits of memory. We also consider another variant of CSP, where given oracle access to two random and independent functions $f_1, f_2 : [N] \rightarrow [N]$ the goal is to find C distinct ordered colliding pairs between f_1 and f_2 (i.e., $(i_1, i_2) \in [N]^2$ such that $f_1(i_1) = f_2(i_2)$) using S bits of memory.

The best known algorithm for the collision search problem is *parallel collision search* (PCS) which was published by van Oorschot and Wiener [32] and has found numerous applications in cryptanalysis (such as space-efficient attacks on double and triple encryption and various dedicated meet-in-the-middle attacks). PCS obtains the time-space tradeoff $T^2 \cdot S = \tilde{O}(C^2 \cdot N)$ (where $S = \tilde{O}(C)$) for both variants of CSP (where $\tilde{O}$ hides a poly-logarithmic factor in N).

Given the importance of PCS, it is natural to ask whether its time-space tradeoff is optimal. First, for $S \approx C$, the answer is clearly positive (ignoring poly-logarithmic factors). This follows by simple probabilistic analysis, as evaluating a random function on T inputs is unlikely to yield more than about T^2/N collisions (i.e., $C = \tilde{O}(T^2/N)$, or $T^2 \cdot S = \tilde{\Omega}(C^2 \cdot N)$). Therefore, the optimality of the tradeoff is not straightforward only when $S \ll C$, as in this range the limited amount of space comes into play.

1.1 Applications of the Collision Search Problem for $S \ll C$

The parameter regime $S \ll C$ for CSP may not seem very interesting at first sight, as it implies that the algorithm has to produce more collisions than it is able store in memory. However, in many CSP applications, we are only interested in a particular collision (referred to as the "golden collision" in [32]) and therefore do not need to store the collisions produced. Thus, the range of parameters where $S \ll C$ is, in fact, very important. Below, we demonstrate the relevance of CSP in case $S \ll C$ for the classical problem of breaking double encryption.

In the double encryption setting, the adversary obtains plaintext-ciphertext pairs (p_i, c_i) for $i \in \{1, 2, \ldots\}$, where $c_i = E2_{k_2}(E1_{k_1}(p_i))$ for block ciphers[1] $E1, E2 : [N] \to [N]$ and key $(k_1, k_2) \in [N]^2$. The adversary focuses on (p_1, c_1) and defines the functions $f_1, f_2 : [N] \to [N]$ as

$$f_1(k) = E1_k(p_1), \text{ and } f_2(k) = E2_k^{-1}(c_1).$$

Then, the adversary applies a collision search algorithm in order to obtain collisions between f_1 and f_2. Each output collision gives a candidate for the key pair (k_1, k_2), which is then verified against the remaining plaintext-ciphertext pairs. Note that there is no need to store a wrong key pair. Since f_1 and f_2 are expected to have N different colliding pairs, after obtaining about N collisions between them, the adversary recovers the correct key pair (or the "golden collision") with high probability.

Consequently, we can use an algorithm for the collision search problem with $C \approx N$ in order to break double encryption given an arbitrary value of S. Plugging $C = N$ into the PCS time-space tradeoff curve gives $T^2 \cdot S = \tilde{O}(N^3)$, which is the currently best known time-space tradeoff for breaking double encryption for any value of S.

Besides breaking double encryption, there are numerous additional applications that are also based on finding a "golden collision" which gives a solution to

[1] We assume for the sake of simplicity that the key and block sizes are equal.

the problem for appropriately defined functions f_1, f_2. A partial list of these applications includes breaking triple encryption [32], and more generally, breaking multiple encryption [17], solving the subset sum problem [16, 17] and solving the generalized birthday problem [33]. Furthermore, in various (more specialized) settings one can reduce the task of breaking a concrete symmetric-key primitive to a meet-in-the-middle procedure which can then be reduced to an instance of the collision search problem with $S \ll C$.

1.2 Optimality of the Known Time-Space Tradeoff for the Collision Search Problem

The importance of CSP (in case $S \ll C$) motivates the question of whether the best known time-space tradeoff for it $T^2 \cdot S = \tilde{O}(C^2 \cdot N)$ can be improved. In this paper, we give a strong negative answer to this question. More specifically, we prove that any algorithm that outputs C distinct collisions in a random function $f : [N] \to [N]$ (or between two independent random functions) with probability at least N^{-2}, satisfies the time-space tradeoff $T^2 \cdot S = \tilde{\Omega}(C^2 \cdot N)$ (where $\tilde{\Omega}$ hides poly-logarithmic factors in N). As an example, given space of $S = 10 \log N$ bits, our lower bound implies that finding $C = N/4$ collisions in f requires time complexity $T = \tilde{\Omega}(N^{3/2})$. On the other hand, prior to this work, even an algorithm with time complexity $T = N$ for the problem could not be ruled out.

An immediate consequence of this result is that the time-space tradeoffs for the cryptanalytic applications mentioned in Section 1.1 cannot be improved by a more efficient collision search procedure (as long as the underlying functions to which collision search is applied are modeled as random functions).

In order to obtain our bound, we use the R-way branching program model of computation, which is a standard non-uniform model for analyzing time-space tradeoffs [11]. In this model, a general approach for proving time-space tradeoff lower bounds was introduced by Borodin and Cook [11]. Since its introduction, this technique has been used to prove time-space tradeoff lower bounds for several problems such as sorting [7, 11], matrix multiplication and fast Fourier transform [36], and universal hashing [25].

We adapt the approach of [11] to the collision search problem, which seems to be its first application in the domain of cryptography. More specifically, we divide a branching program for CSP with running time T into L short time intervals of length T/L, and prove that in each such interval, the program cannot make a lot of *progress* towards outputting the desired number of C collisions. In particular, the probability that the program (starting its computation from any specific memory state) outputs C/L collisions in an interval is minuscule. Finally, a union bound over all possible memory states establishes the result. Our adaptation requires deeper insight about the collision search problem and a careful choice of parameters.

1.3 Time-Space Tradeoffs in Various Models of Computation

Given the optimality of best known time-space tradeoff for CSP, one may wonder whether we can prove the optimality of the best known tradeoffs for its applications (such as the ones mentioned in Section 1.1). Unfortunately, we give strong evidence that proving such a result may be very difficult and would overcome a variant of a long-standing barrier in complexity theory.

The property of the collision search problem that enables proving tight time-space tradeoff lower bounds for it, is that its output length is (about) C words. Since it is possible to find C collisions in f for a small value of C with negligible space and nearly-optimal time complexity (e.g., by Floyd's algorithm [24]), time-space tradeoffs for such parameters are already known to be tight. Hence, we may assume that the output length C is large (namely $C = N^{\alpha}$ for some $\alpha > 0$). In contrast, the output length in the above applications is very short (e.g., in breaking double encryption, the output is a short key). The challenge of proving strong time-space tradeoff lower bounds for such short-output problems with polynomial-time algorithms[2] is open since the study of such tradeoffs was introduced in 1966 by Cobham [15] (and it has been subject of extensive research [10]).[3] In particular, the technique for proving time-space lower bounds of [11] (that we adapt here to CSP) is inapplicable to short-output problems since it is not clear how to measure the progress of an algorithm towards solving such a problem.

While the barrier of proving time-space lower bounds for short-output problems seems very difficult to overcome, it only applies to general (unrestricted) models of computation. On the other hand, under more restricted computational models, very strong time-space lower bounds are known for various problems. One such restricted model is the streaming model. Here, the input is a stream of elements that can be read sequentially, but a (single-pass) streaming algorithm cannot access a previous element of the stream, unless it is stored in memory. The streaming model is subject to active research in general (see [2], and [30] for a recent result), and specifically in the area of cryptography (cf., [13, 22, 27]).

The pebbling model is another well-studied restricted model of computation in which strong time-space tradeoff lower bounds are known [29]. In this model, a pebbling game is played on a specific circuit realizing the function considered. The circuit is viewed as a directed acyclic graph (DAG) and the goal is to pebble each of the output nodes given a limited number of pebbles (which model words of memory). The main rule of the pebbling game is that a non-input node can be pebbled only if all its processors are pebbled. In the context of cryptography, the

[2] The input length of a problem with access to some function $f : [M] \to [N]$ is the number of bits required to represent the function, namely, $N \log M$. With this encoding, it is clear, for example, that breaking double encryption can be done in polynomial time in the input length.

[3] For general problems in NP, stronger time-space tradeoff lower bounds are known in the uniform setting for problems such as SAT (cf. [20]). However, as far as we know, these lower bounds are very loose and do not seem to be relevant to cryptography.

pebbling model has played a significant role in the design and analysis memory-hard functions [3, 4, 19].

We also mention comparison-based models where the algorithm does not have direct access to the internal representation of the elements of the input, but can only compare them in pairs. A well-known time-space lower bound in such models was obtained by Yao for the *element distinctness problem* [35], where the goal is to determine whether there exist two identical elements in an array of length N. While Yao's bound $T \cdot S = \Omega(N^{2-o(1)})$ for element distinctness is almost tight in comparison-based models, the algorithm of Beame et al. [8] showed how to beat this bound and obtained $T^2 \cdot S = O(N^3)$ by working outside the restricted model (i.e., exploiting the internal representations of the elements). Recently, Tessaro and Thiruvengadam showed bi-directional space-tight reductions between breaking double encryption and solving the element distinctness problem [31]. In fact, the algorithm of [8] is a variant of PCS which obtains a similar tradeoff for breaking double encryption.

Finally, a different restricted model of computation for proving time-space tradeoff lower bounds in the domain of cryptography was proposed in [6] by Barkan, Biham and Shamir. This work analyzed the problem of inverting a random function $f : [N] \to [N]$ with preprocessing, and proposed a model that generalizes the algorithms of Hellman [21] and Oechslin [28] for the problem. The main result of [6] was a proof that any algorithm in their model cannot substantially improve upon the known time-space tradeoffs for the function inversion problem.

1.4 The Post-Filtering Model of Computation

Even though restricted computational models are natural choices for analyzing a variety of problems, none of the existing ones seems relevant to obtaining time-space lower bounds for the cryptanalytic problems described in Section 1.1. For example, the streaming model is inapplicable to cryptanalysis of double encryption, since the best known key-recovery attack requires only two plaintext-ciphertext pairs (and it does not make sense to restrict access to the block cipher itself). The pebbling model is irrelevant since it is not clear which circuit should be pebbled. Considering comparison-based models, one can apply Yao's bound for element distinctness to the problem of breaking double encryption (as implied by the result of Tessaro and Thiruvengadam [31]). However, PCS beats Yao's bound, which implies that the model is too restrictive for this specific problem. Finally, the precomputation setting analyzed in [6, 21, 28] is quite different from ours and the tools used in its analysis seem inapplicable in our context.

Nevertheless, we would still like to obtain some meaningful time-space lower bounds that apply to basic cryptanalytic problems. Hence, we put forward a new restricted model of computation, which we call the *post-filtering model*, where the algorithm obtains full access to a *part* of the input, while access to the remaining part is replaced with access to a *post-filtering oracle*.

The post-filtering model may seem to have little to do with space complexity, yet it allows proving time-space lower bounds. The reason for this is that in

some problems the input can be partitioned such that given the first part, there are many equally-likely potential solutions. Consequently, an algorithm with full access to (only) the first part of the input has to produce many potential outputs to be post-filtered by the oracle. Thus, the model forces a reduction from a problem with a short output to a related problem with a long output, for which time-space tradeoff lower bounds are provable with existing techniques.

In the post-filtering model, we focus on double encryption, as it seems to be the most fundamental problem listed in Section 1.1. Indeed, it is the basis for many other more involved meet-in-the-middle type of attacks (such as breaking triple encryption and specialized attacks on concrete symmetric-key constructions). In our analysis, we give the adversary full access to (p_1, c_1) and the block ciphers $E1, E2$, while access to the remaining plaintext-ciphertext pairs is replaced with access to a post-filtering oracle that filters out wrong key guesses. We exploit the fact that there are many possible (k_1, k_2) pairs that are consistent with (p_1, c_1), and prove that any post-filtering attack on double encryption that succeeds with constant probability satisfies the time-space tradeoff curve $T^2 \cdot S = \tilde{\Omega}(N^3)$. This matches the performance of the best known attack that uses PCS.

Technically, we obtain this result based on the optimality of the tradeoff for CSP, but it is a conceptually stronger result, as the post-filtering model abstracts away the (lower-level) collision search problem. The optimality of the tradeoff for double encryption in the post-filtering model implies that if an improved algorithm exists, then it must deviate from the post-filtering model by simultaneously combining information from several plaintext-ciphertext pairs in a meaningful way. This can be viewed both as a barrier, but also as an opportunity for improvement.

We also mention that a different approach to obtaining meaningful results for short-output problems (despite the aforementioned barrier) is to use space-efficient reductions in order to prove relations among tradeoffs for these problems [5]. The recent work by Tessaro and Thiruvengadam [31] (that showed reductions between attacking double encryption and solving the element distinctness problem) is a relevant example of this approach. We note that it is possible to adapt the post-filtering model to the element distinctness problem, and obtain a similar bound to the one we obtain for double encryption.

1.5 Paper Organization

The rest of the paper is structured as follows. We describe preliminaries in Section 2. Then, we prove time-space tradeoff lower bounds for collision search in a single function and between two functions in Section 3 and Section 4, respectively. We discuss relevant barriers in complexity theory in Section 5 and prove a time-space tradeoff lower bound for double encryption in the post-filtering model in Section 6. Finally, we conclude the paper in Section 7.

2 Preliminaries

Let N be a natural number and denote $[N] = \{0, 1, \ldots, N-1\}$. We use the standard $\tilde{O}$ and $\tilde{\Omega}$ notations that suppress poly-logarithmic factors in N.

Let X be a finite set. We write $x \xleftarrow{\$} X$ to indicate that x is a random variable sampled uniformly from X. We denote by $x \leftarrow \mathcal{D}$ a random variable x sampled according to the distribution $\mathcal{D}$.

We use a weak version of Stirling's approximation, which asserts that $n! > (n/e)^n$ for every positive integer n.

In this paper, we are interested in counting distinct collision pairs. A colliding pair in a function $f : [N] \to [N]$ is an unordered pair of indices $(i_1, i_2) \in [N]^2$ such that $i_1 \neq i_2$ and $f(i_1) = f(i_2)$. When considering two functions $f_1, f_2 : [N] \to [N]$, a colliding pair between f_1 and f_2 is an ordered pair $(i_1, i_2) \in [N]^2$ such that $f_1(i_1) = f_2(i_2)$. Two pairs (i_1, i_2) and (j_1, j_2) and *disjoint* if they do not share any index.

Let $f : [N] \to [N]$ be a function. For an integer $t \geq 2$, a t-way collision in f is an (unordered) set of t distinct indices $\{i_1, \ldots, i_t\}$ such that $f(i_1) = f(i_2) = \ldots = f(i_t)$. Note that a t-way collision in f contains $t(t-1)/2$ distinct colliding pairs, but can only be partitioned into $\lfloor t/2 \rfloor$ disjoint colliding pairs.

We may refer to a function $f : [N] \to [N]$ as a vector (or a string) in the space $[N]^N$, and visa-versa. More specifically a string $x \in [N]^N$ represents a function $f_x : [N] \to [N]$ defined as $f_x(i) = x[i]$. In this paper, we will switch between these representations. For example, we define a collision in $x \in [N]^N$ as an unordered pair of indices $(i_1, i_2) \in [N]^2$ such that $i_1 \neq i_2$ and $x[i_1] = x[i_2]$.

2.1 The Collision Search Problem

Let N, C be positive integer parameters. Given random access to a function $f : [N] \to [N]$, the goal in the collision search problem CSP(C) is to output a multi-set of pairs $(i_1^{(j)}, i_2^{(j)}) \in [N]^2$ for $j \in \{1, 2, \ldots\}$ (at any order) such that each pair is colliding (i.e., $f(i_1^{(j)}) = f(i_2^{(j)})$ but $i_1^{(j)} \neq i_2^{(j)}$) and the multi-set contains at least C distinct (unordered) colliding pairs.

CSP can be extended to two functions $f_1, f_2 : [N] \to [N]$. Here, the goal is to output at least C distinct ordered colliding pairs between the functions.

We define CSP such that all distinct colliding pairs are accounted for, as most CSP applications (such as breaking double encryption) are interested in each such pair. Of course, if we under-count pairs induced by t-way collisions (e.g., by considering only disjoint pairs), we can obtain a (slightly) better lower bound.

In Appendix A we summarize the PCS algorithm that solves CSP(C) and obtains the time-space tradeoff $T^2 \cdot S = \tilde{O}(C^2 \cdot N)$.

2.2 *R*-Way Branching Programs

The model of R-way branching programs is a very general and powerful non-uniform model of computation, introduced in [11]. An R-way branching pro-

gram is a directed acyclic graph in which each node represents a memory state of the program. At each node, a single input variable is queried and the program branches to the next state according to the value of this variable (possibly printing an output value along the way). Thus, a path in the graph of length T represents T time steps (input variable queries) of the program.

Let $R \geq 2$ be an integer. Formally, an R-way branching program $\mathcal{P}$ on an input vector consisting of N input variables $x = x[1], \ldots, x[N] \in [R]^N$ is a directed acyclic graph with a single source node, such that every non-sink node has out-degree R and is labeled with an index $i \in [N]$ corresponding to an input variable to be queried. Every edge is labeled with an element of $[R]$ (corresponding to the value of the queried variable) such that no edge (u, v) and (u, w) where $v \neq w$ share a label. Furthermore, every vertex v is associated with an instruction to print a (possibly empty) value $p(v)$.

The computation path of $\mathcal{P}$ on input x is the (unique) path $\pi = (v_0, v_1), (v_1, v_2), \ldots, (v_{\ell-1}, v_\ell)$ such that v_0 is the source node, v_ℓ is a sink node, and for all $i \in \{0, \ldots, \ell - 1\}$, if v_i is labeled j, then the label of (v_i, v_{i+1}) is equal to $x[j]$. We denote by $\mathcal{P}(x)$ the output of $\mathcal{P}$ on input x. It is defined as the concatenation of the values printed by the vertices along the computation path of $\mathcal{P}$ on input x, namely, $p(v_0)p(v_1)\ldots p(v_\ell)$.

The *height* of an R-way branching program $\mathcal{P}$ is the length of the longest path in its graph, and its *size* is its number of vertices. The *time complexity* of a branching program is defined as its length, while the *space* used by a branching program is defined as the log of its size. Note that the input and output are not counted towards the space used by the branching program, and time complexity is measured only in terms of the number of queries to the input variables.

An R-way branching program is called *levelled* if its nodes are assigned levels such that the source is in level 0 and the out-edges of each node at level k go only to nodes at level $k + 1$. It is shown in [12] that any branching program of size 2^S can be converted into an equivalent levelled branching program with the same length and size of at most 2^{2S}.

2.3 Our Model of Computation

We use the R-way branching program model in order to prove the time-space lower bound for collision search in Section 3. However, in sections 4 and 6 it will be more natural to consider algorithms rather than branching programs. Nevertheless, the model of computation that we use in these sections is equivalent to the branching program model. More specifically, we consider an algorithm $\mathcal{A}$ with access to input variables in $[R]$ (for a value of R depending on the problem). The algorithm has space of S bits, where the input and output do not count towards the space complexity. Each query of $\mathcal{A}$ to one of its input variables costs one time unit, but other operations (such as reading or writing to memory) are for free. This makes our computational model (and our lower bounds) very strong. Note that an algorithm $\mathcal{A}$ in our model is equivalent to an R-way branching program $\mathcal{P}$ with the same time and space complexities.

***R*-Way Branching Programs for Collision Search** We model the algorithm for CSP(C) as an N-way branching program $\mathcal{P}$ of height T and size 2^S. We assume that $\mathcal{P}$ is deterministic, but our lower bounds extend to randomized branching programs as well by Yao's minimax principle [34].

For convenience, we denote by $|\mathcal{P}(x)|$ the number of distinct colliding pairs (ignoring duplicates) output by $\mathcal{P}$ on input $x \in [N]^N$, which represents a function $f_x : [N] \to [N]$ defined as $f_x(i) = x[i]$. We define $|\mathcal{P}(x)| = -1$ if $\mathcal{P}(x)$ is erroneous, i.e., it outputs a pair which does not collide on x.

Input Representation In the domain of complexity theory, if an algorithm $\mathcal{A}$ (or branching program $\mathcal{P}$) has random access to $x \in [N]^N$, then x is typically treated as an input, using the notation $\mathcal{A}(x)$ (or $\mathcal{P}(x)$). We use such notation in sections 3, 4 and 5. On the other hand, in cryptography, similar random access of an algorithm $\mathcal{A}$ to a function $f_x : [N] \to [N]$ is typically modeled by viewing $f = f_x$ as an oracle and using the notation $\mathcal{A}^f$. We use this notation in Section 6.

3 A Time-Space Tradeoff Lower Bound for Collision Search in a Function

In this section, we prove the following theorem.

Theorem 1. *Let N, C, T, S be positive integer parameters such that $N > 8$, $S \geq 5(\log^2 N + \log N)$ and $C \leq N/4$. If an N-way branching program $\mathcal{P}$ for CSP(C) of height T and size 2^S satisfies*

$$\Pr_x[|\mathcal{P}(x)| \geq C] \geq N^{-2}, \text{ where } x \xleftarrow{\$} [N]^N, \text{ then}$$

$$T^2 \cdot S \geq \frac{1}{(6e \cdot \log N)^2} \cdot C^2 \cdot N.$$

Remark 1. If $S < 5(\log^2 N + \log N)$, we can apply the theorem with $S' = 5(\log^2 N + \log N)$. If $C > N/4$, we can apply the theorem with $C' = N/4$ (note that with high probability a random function only contains $O(N)$ collisions). In both cases the loss in the bound is small.

Remark 2. The theorem is formulated for deterministic algorithms (or branching programs). However, by Yao's minimax principle it also applies to randomized algorithms, which are viewed as distributions over deterministic algorithms. In this case, the probability is also taken over the randomness of the algorithm.

3.1 Overview of the Proof

The proof is an adaptation of the general approach of [11]. This adaptation requires additional insight which we summarize below.

We first prove that every shallow branching program of a small height T' outputs C' (distinct) collisions in a random input x with negligible probability,

denoted here by $\epsilon = \epsilon(T', C')$ (for appropriate values of T' and C'). Note that this proof has nothing to do with space complexity.

Then, given a branching program $\mathcal{P}$ of size 2^S and height T, we level it to obtain a levelled branching program $\mathcal{P}'$, losing a factor of 2 in its space S. We then split $\mathcal{P}'$ into L layers (for a carefully chosen value of L), each of height at most $T' = T/L$. In order to produce C collisions, at least one such layer has to produce $C' = C/L$ collisions, namely, there exists a subprogram of $\mathcal{P}'$ (defined by its source node) of length T' in some layer that outputs C' collisions. Since $\mathcal{P}'$ contains 2^{2S} such subprograms, we take a union bound over all of them and conclude that the probability in which $\mathcal{P}'$ outputs C (distinct) collisions is upper bounded by $2^{2S} \cdot \epsilon$.

With T' queries to the input x, the expected number of encountered collisions is about $(T')^2/N$, and we would like to prove a strong concentration inequality which shows that it is extremely unlikely to encounter $C' = c \cdot (T')^2/N$ collisions for sufficiently small $c > 1$, thus obtaining a strong upper bound on ϵ. Indeed, in order to obtain a meaningful upper bound on the success probability of $\mathcal{P}'$, we need $\epsilon \ll 2^{-2S}$.

The above calculation justifies the relation between C' and T'. It remains to choose L such that we can indeed prove that $\epsilon \ll 2^{-2S}$ and obtain the desired bound $T^2 \cdot S = \tilde{\Omega}(C^2 \cdot N)$. Some calculation shows that for the sake of obtaining a tight bound, we need to choose L such that C' is a bit larger than the space $2S$. Intuitively, a choice of L such that C' is much larger than $2S$ will artificially allow the program to output too many collisions (much more than its space) in limited time and result in a loose bound. On the other hand, if we choose L for which C' is smaller than $2S$, $\mathcal{P}'$ will actually be able to output C' collisions with sufficiently high probability and we will not be able to obtain the required upper bound on ϵ such that $\epsilon \ll 2^{-2S}$.

Hence, the constraint $\epsilon \ll 2^{-2S}$ translates into $\epsilon = 2^{-\tilde{\Omega}(C')}$. In other words, we need to prove a concentration inequality that decays exponentially with the number of collisions per layer C'. Unfortunately, obtaining such a strong bound on ϵ is impossible in general. For example, suppose that $T' = N^{3/4}$, hence $C' \approx N^{1/2}$. Note that $C' \approx N^{1/2}$ distinct colliding pairs can be obtained via a t-way collision for $t \approx N^{1/4}$. The probability of obtaining such a t-way collision is at least

$$N^{-N^{1/4}} = 2^{-\log N \cdot N^{1/4}} \gg 2^{-N^{1/2}} \approx 2^{-C'}.$$

Fortunately, most functions do not contain such a large t-way collision (as proved in Lemma 1), and for these functions, we can prove the required bound on ϵ. Restricting our attention to such functions allows the proof to go through.

We note that the poly-logarithmic factors in Theorem 1 can be improved. In particular, a refined version of Lemma 1 (which would consider t-way collisions for various values of t, rather than merely $t = 3 \log N$) would yield such an improvement. However, in this paper we opt for simplicity at the expense of low-level optimizations.

Finally, we also mention the related work by Chakrabarti and Chen [14] which analyzed time-space tradeoffs for the *memory game* with cards. This game

is played with N distinct pairs of cards laid face-down, and the goal is to output all "colliding" pairs. Although the memory game resembles CSP (and the lower bound obtained in [14] is similar to ours), its analysis in [14] does not seem to apply to CSP. This is mainly due to t-way collisions for $t > 2$ which are possible in the collision search problem, but not in the memory game where the cards are composed of distinct pairs.

3.2 Bounding the Number of Collisions Output by Shallow Branching Programs

Lemma 1. *Let $t > 0$ be an integer. Then,*

$$\Pr_x[x \text{ contains a } t\text{-way collision}] \leq \frac{N}{t!},$$

where $x \xleftarrow{\$} [N]^N$. In particular, for $N > 8$,

$$\Pr_x[x \text{ contains a } 3\log N\text{-way collision}] \leq \frac{N}{(3\log N)!} \leq N \cdot \left(\frac{e}{3\log N}\right)^{3\log N} \leq N^{-3}.$$

Proof. Fix a set of t distinct indices in $[N]$, $\{i_1, \ldots, i_t\}$. We have

$$\Pr[x[i_1] = x[i_2] = \ldots = x[i_t]] = N^{-t+1}.$$

The number of such sets of indices is $\binom{N}{t} \leq \frac{N^t}{t!}$. Taking a union bound over all sets,

$$\Pr[x \text{ contains a } t\text{-way collision}] \leq \frac{N^t}{t!} \cdot N^{-t+1} = \frac{N}{t!}.$$

■

Lemma 2. *For all T', C' and $N > 8$, any N-way branching program $\mathcal{P}$ of height at most T' satisfies*

$$\Pr_x[|\mathcal{P}(x)| \geq C' \mid x \text{ does not have a } 3\log N\text{-way collision}] \leq$$
$$2 \cdot N^{3\log N} \cdot \left(\frac{3e \cdot \log N \cdot (T')^2}{C' \cdot N}\right)^{C'/6\log N}, \text{ where } x \xleftarrow{\$} [N]^N.$$

Proof. Denote by $\mathcal{E}$ the event that x does not have a $3\log N$-way collision. Suppose that x satisfies this condition and assume that C' distinct colliding index pairs are output by $\mathcal{P}(x)$. Assume that these pairs form k disjoint sets, where set i is of size t_i for $2 \leq t_i < 3\log N$, and gives a t_i-way collision. Since a t-way collision results in $t(t-1)/2$ distinct colliding pairs, we have $\sum_{i=1}^{k} t_i(t_i-1)/2 \geq C'$. On the other hand, each set of size t_i can be partitioned into $\lfloor t_i/2 \rfloor$ disjoint colliding pairs. Altogether, the C' distinct colliding index pairs can be partitioned into

$$\sum_{i=1}^{k} \lfloor t_i/2 \rfloor \geq \sum_{i=1}^{k} (t_i - 1)/2 \geq \frac{1}{3\log N} \cdot \sum_{i=1}^{k} t_i(t_i-1)/2 \geq \frac{C'}{3\log N}$$

disjoint colliding pairs.

Using the fact that $\Pr_x[\mathcal{E}] \geq N^{-3\log N}$, we obtain

$$\Pr_x[|\mathcal{P}(x)| \geq C' \mid \mathcal{E}] \leq$$
$$\Pr_x\left[\mathcal{P}(x) \text{ is correct and outputs } \frac{C'}{3\log N} \text{ disjoint colliding pairs} \mid \mathcal{E}\right] \leq$$
$$\Pr_x\left[\mathcal{P}(x) \text{ is correct and outputs } \frac{C'}{3\log N} \text{ disjoint colliding pairs}\right] / \Pr_x[\mathcal{E}] \leq$$
$$N^{3\log N} \cdot \Pr_x\left[\mathcal{P}(x) \text{ is correct and outputs } \frac{C'}{3\log N} \text{ disjoint colliding pairs}\right].$$

It remains to prove that

$$\Pr_x[\mathcal{P}(x) \text{ is correct and outputs } \frac{C'}{3\log N} \text{ disjoint colliding pairs}] \leq$$
$$2\left(\frac{3e\cdot\log N\cdot(T')^2}{C'\cdot N}\right)^{C'/6\log N}.$$

For $K = \frac{C'}{6\log N}$, denote by $\mathcal{E}_1$ the event that $\mathcal{P}(x)$ queries K disjoint colliding index pairs and by $\mathcal{E}_2$ the event that $\mathcal{P}(x)$ is correct and outputs K disjoint colliding index pairs such that in each one, at least one index is not queried. Note that if $\mathcal{P}(x)$ is correct and outputs $\frac{C'}{3\log N}$ disjoint colliding pairs, then either $\mathcal{E}_1$ or $\mathcal{E}_2$ occurs.

For a fixed set of disjoint K pairs of query indices, the probability that they all collide on a uniform input x is N^{-K}. The number of ways to select such K query index pairs from a set of T' index queries is

$$\binom{T'}{2K}\cdot\frac{(2K)!}{2^K\cdot K!} \leq \frac{(T')^{2K}}{2^K\cdot(K/e)^K} = \left(\frac{(T')^2}{2/e\cdot K}\right)^K.$$

Taking a union bound over all sets of K query index pairs, we conclude

$$\Pr[\mathcal{E}_1] \leq \left(\frac{(T')^2}{2/e\cdot K}\right)^K \cdot N^{-K}.$$

In addition,

$$\Pr[\mathcal{E}_2] = N^{-K}.$$

Finally,

$$\Pr_x[\mathcal{P}(x) \text{ is correct and outputs } \frac{C'}{3\log N} \text{ disjoint colliding pairs}] \leq$$
$$\Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] \leq \left(\frac{(T')^2}{2/e\cdot K}\right)^K \cdot N^{-K} + N^{-K} \leq 2\left(\frac{3e\cdot\log N\cdot(T')^2}{C'\cdot N}\right)^{C'/6\log N},$$

as claimed. ■

3.3 Proof of Theorem 1

Proof (of Theorem 1). Let $\mathcal{P}$ be a branching program of height T and size 2^S. We prove the contrapositive statement of the theorem by assuming

$$T^2 \cdot S < \frac{1}{(6e \cdot \log N)^2} \cdot C^2 \cdot N, \tag{1}$$

and showing that

$$\Pr_x[|\mathcal{P}(x)| < C] \geq 1 - N^{-2}.$$

Denote by $\mathcal{E}$ the event that x does not have a $3 \log N$-way collision. We lower bound the failure probability of $\mathcal{P}$ by

$$\Pr_x[|\mathcal{P}(x)| < C] \geq \Pr[\mathcal{E}] \cdot \Pr[|\mathcal{P}(x)| < C \mid \mathcal{E}] = (1 - \Pr[\neg\mathcal{E}]) \cdot \Pr[|\mathcal{P}(x)| < C \mid \mathcal{E}] \geq \Pr[|\mathcal{P}(x)| < C \mid \mathcal{E}] - \Pr[\neg\mathcal{E}].$$

In the following, we prove based on Lemma 2 that

$$\Pr_x[|\mathcal{P}(x)| \geq C \mid \mathcal{E}] \leq N^{-3}. \tag{2}$$

Therefore, combining (2) with Lemma 1,

$$\Pr_x[|\mathcal{P}(x)| < C] \geq \Pr[|\mathcal{P}(x)| < C \mid \mathcal{E}] - \Pr[\neg\mathcal{E}] \geq 1 - N^{-3} - N^{-3} \geq 1 - N^{-2},$$

as required.

It remains to prove (2). We first level the branching program to obtain a levelled branching problem $\mathcal{P}'$ of length T and size at most 2^{2S}. Partition $\mathcal{P}'$ into $L = \frac{T}{\sqrt{S \cdot N}}$ layers, each of height at most $T' = T/L$. By an averaging argument, if $\mathcal{P}'$ outputs C distinct colliding pairs, then there exists a layer that outputs at least $C' = C/L$ distinct colliding pairs.[4] Hence, the probability that $\mathcal{P}'$ outputs at least C distinct colliding pairs is upper bounded by the probability that some layer outputs C' pairs. Since there are at most 2^{2S} subprograms (each defined by its source node) in $\mathcal{P}'$, by Lemma 2 and a union bound over all subprograms we obtain

$$\begin{aligned}
&\Pr_x[|\mathcal{P}'(x)| \geq C \mid \mathcal{E}] \leq \\
&2^{2S} \cdot 2 \cdot N^{3 \log N} \cdot \left(\frac{3e \cdot \log N \cdot (T')^2}{C' \cdot N}\right)^{C'/6 \log N} = \\
&2^{2S} \cdot 2 \cdot N^{3 \log N} \cdot \left(\frac{3e \cdot \log N \cdot (T/L)^2}{C/L \cdot N}\right)^{C'/6 \log N} = \\
&2^{2S} \cdot 2 \cdot N^{3 \log N} \cdot \left(\frac{3e \cdot \log N \cdot T^2}{C \cdot L \cdot N}\right)^{C'/6 \log N} = \\
&2^{2S} \cdot 2 \cdot N^{3 \log N} \cdot \left(\frac{3e \cdot \log N \cdot T \cdot \sqrt{S}}{C \cdot \sqrt{N}}\right)^{C'/6 \log N}.
\end{aligned}$$

[4] Note that $C' = C/L = \frac{C \cdot \sqrt{S \cdot N}}{T} \approx S$, as suggested in the overview at the beginning of Section 3.

If $\frac{3e \cdot \log N \cdot T \cdot \sqrt{S}}{C \cdot \sqrt{N}} > 1/2$, then $T^2 \cdot S > \frac{1}{(6e \cdot \log N)^2} \cdot C^2 \cdot N$, in contradiction to (1). Therefore,

$$2^{2S} \cdot 2 \cdot N^{3 \log N} \cdot \left(\frac{3e \cdot \log N \cdot T \cdot \sqrt{S}}{C \cdot \sqrt{N}} \right)^{C'/6 \log N} \leq$$
$$2^{2S} \cdot 2 \cdot N^{3 \log N} \cdot 2^{-C'/6 \log N} = 2^{2S+1+3 \log^2 N - \frac{1}{6 \log N} \cdot C \cdot \sqrt{S \cdot N}/T}.$$

According to (1),

$$\frac{1}{6 \log N} \cdot \frac{C \cdot \sqrt{N}}{T} \geq e \cdot \sqrt{S}.$$

Hence,

$$2^{2S+1+3 \log^2 N - \frac{1}{6 \log N} \cdot C \cdot \sqrt{S \cdot N}/T} \leq 2^{2S+1+3 \log^2 N - e \cdot S} = 2^{S(2-e)+1+3 \log^2 N} \leq N^{-3}$$

(since $S \geq 5(\log^2 N + \log N)$), concluding the proof.

■

4 A Time-Space Tradeoff Lower Bound for Collision Search between Two Functions

In this section, we analyze the problem of collision search between two independent and random functions. For convenience, we consider algorithms rather than branching programs, even though they are equivalent in the computational model we consider.

Theorem 2. *Let N, C, T, S be positive integer parameters such that $S \geq 5(\log^2 N + \log N)$, $C \leq N$ and $N > 4$. Let $\mathcal{A}$ be an algorithm that outputs C colliding pairs between two independent random functions $f_1, f_2 : [N] \to [N]$ with probability at least N^{-2}, using T queries to f_1 and f_2 and space of S bits. Then, $\mathcal{A}$ satisfies the time-space tradeoff lower bound*

$$T^2 \cdot S \geq \frac{1}{(24e \cdot \log N)^2} \cdot C^2 \cdot N.$$

We note that as Theorem 1, Theorem 2 also applies to randomized algorithms.

It is possible to prove Theorem 2 by using the same technique that was used to prove Theorem 1 (in fact, this results in slightly better parameters). Instead, we give a simpler proof by a reduction from the problem of collision search in a single function, under the mild assumption that the output length of $\mathcal{A}$ (i.e., the total number of elements of $[N]$ that it outputs) is not larger than its number of queries T. This assumption is not needed in general.

Proof. We reduce the problem of outputting C collisions between two independent random functions $f_1, f_2 : [N] \to [N]$ from the problem of outputting $C' = C/2$ collisions in a single random function $f : [2N] \to [2N]$.

Let $\mathcal{A}$ be an algorithm for finding colliding pairs between two functions with domain and range $[N]$. Let $f : [2N] \to [2N]$ be a random function. We devise an algorithm $\mathcal{A}'$ that outputs $C' = C/2$ collisions in f as follows. Define $f_1, f_2 : [N] \to [N]$ as

$$f_1(i) = f(i) \bmod N, \text{ and}$$
$$f_2(i) = f(i+N) \bmod N.$$

It is easy to verify that f_1, f_2 are two independent random functions.

The algorithm $\mathcal{A}'$ runs $\mathcal{A}$ with parameter C, giving it access to f_1, f_2. For every pair (i_1, i_2) such that $f_1(i_1) = f_2(i_2)$ output by $\mathcal{A}$, algorithm $\mathcal{A}'$ checks whether $f(i_1) = f(i_2 + N)$, and if so, outputs the pair $(i_1, i_2 + N)$. Since we assume that the output length of $\mathcal{A}$ is not larger than its number of queries T, we have $T' \leq 2T$. Moreover, the space used by $\mathcal{A}'$ is essentially the same as that of $\mathcal{A}$ (in particular , $S' \leq 2S$).

We call a function $f : [2N] \to [2N]$ *bad* if $\mathcal{A}$ outputs at least C collisions on f_1, f_2 derived from f, but $\mathcal{A}'$ outputs less than $C/2$ collision on f. We call f *good*, if $\mathcal{A}$ outputs at least C collisions on f_1, f_2 derived from f, and $\mathcal{A}'$ outputs at least $C/2$ collision on f. We claim that the number of good functions is at least the number of bad functions. Indeed, let M be a mapping between functions $f : [2N] \to [2N]$ which maps f to $\hat{f} = M(f)$, defined as follows:

$$\hat{f}(i) = \begin{cases} f(i), & \text{for } i < N \\ f(i) - N, & \text{for } i \geq N \text{ and } f(i) \geq N \\ f(i) + N, & \text{for } i \geq N \text{ and } f(i) < N. \end{cases}$$

Note that $\mathcal{A}$ is run with the same input on f and $M(f)$, hence it produces the same output. Moreover, for every $(i_1, i_2) \in [N]^2$, if $f_1(i_1) = f_2(i_2)$ and $f(i_1) \neq f(i_2 + N)$ then $\hat{f}_1(i_1) = \hat{f}_2(i_2)$ and $\hat{f}(i_1) = \hat{f}(i_2 + N)$ (where $\hat{f} = M(f)$). Hence, every bad function is mapped by M to a good function. Finally, $M(M(f)) = f$, implying that M is a permutation on the space of functions, proving that the number of good functions is at least the number of bad functions.

Let X, X' be random variables for the number of distinct number of colliding pairs output by $\mathcal{A}, \mathcal{A}'$, respectively. We have shown that

$$\Pr[X' \geq C/2] \geq 1/2 \cdot \Pr[X \geq C], \text{ or } \Pr[X \geq C] \leq 2 \cdot \Pr[X' \geq C/2].$$

Applying Theorem 1 with $N' = 2N, C' = C/2, S' = 2S, T' = 2T$ concludes the proof. ■

5 Time-Space Complexity Barriers and their Cryptanalytic Variants

In this section, we argue that it may be very difficult to prove tight time-space lower bounds (i.e., analogs of Theorem 1 and Theorem 2) for cryptanalytic problems with short outputs, whose most efficient algorithms seem to require substantial space. The results of this section motivate the restricted post-filtering

model of computation used in Section 6 to prove a time-space tradeoff lower bound for double encryption.

The *smallest fundamental complexity barrier* (as named in [10], and formulated as a challenge) is to find an explicit Boolean decision problem $h : \{0,1\}^n \to \{0,1\}$ in P for which $T \cdot S = O(n \log n)$ is not possible. Since its formulation, the original barrier has been overcome in [9] by Beame et al. which gave explicit examples of problems for which any algorithm for computing them with space of $S = n^{1-\epsilon}$ bits (where $\epsilon > 0$ is an arbitrarily small constant), requires time complexity of at least $T = \Omega(n\sqrt{\log n / \log\log n})$. This time-space tradeoff lower bound was proved for R-way branching programs.

Despite this breakthrough, its does not give any non-trivial lower bound on the space of an algorithm running in time (say) $T = n \log n$. Therefore, the complexity barrier was reformulated in [9] to proving a non-trivial time-space tradeoff lower bound when $T = n(\log n)^{\omega(1)}$.

In order to generalize this barrier to cryptanalytic problems, we consider problems with longer input variables and longer outputs (but still poly-logarithmic in the input length). Consequently, we require that $T \cdot S = \Omega(n^{1+\epsilon})$ for some $\epsilon > 0$.

As a simple example, we consider the problem of finding a 3-way collision in a function, represented by an input $x \in [N]^N$. The goal in this problem is to find 3 distinct indices i_1, i_2, i_3 such that $x[i_1] = x[i_2] = x[i_3]$. Note that the output length is $3 \log N$ and is short (poly-logarithmic in the length of x which is $n = N \cdot \log N$). We can formulate the generalized barrier for this problem as proving that any algorithm requires $T \cdot S = \Omega(N^{1+\epsilon})$. However, in cryptography, we are typically interested in average-case, rather than worst-case problems. In particular, one is typically interested in finding collisions in random functions.

Consider the uniform distribution over $x \in [N]^N$ and a trivial algorithm that evaluates $T = O(N^{2/3})$ arbitrary input variables and looks for a 3-way collision among them. Simple probabilistic analysis shows that the algorithm succeeds with high probability. Such sub-linear algorithms demonstrate that we need to further generalize the challenge above to average-case problems. Below we formulate a challenge for finding a 3-way collision.

Challenge 1 *Prove that there exist $\epsilon > 0$ and $\delta > 0$ such that any algorithm that succeeds in finding a 3-way collision in a uniformly chosen $x \in [N]^N$ for all sufficiently large N with probability at least $3/4$ and $T = N^{2/3+\epsilon}$, satisfies $S \geq N^{\delta}$.*

The difficulty in overcoming Challenge 1 stems from the fact that currently known techniques are not able to prove space lower bounds (of the form $S \geq n^{\delta}$ for $\delta > 0$) for short-output problems with input size n that are solvable in time $\hat{T}$ whenever we allow $T = \hat{T} \cdot n^{\epsilon}$ for some $\epsilon > 0$. In contrast, for problems where the output size is $n^{\Omega(1)}$, such lower bounds are known. Thus, overcoming Challenge 1 would be a breakthrough and perhaps lead towards overcoming a similar barrier for decision problems.

We note that the best known time-space tradeoff algorithm for finding a 3-way collision was published by Joux and Lucks [23] and obtains $T \cdot S = \tilde{O}(N)$

for $T \geq N^{2/3}$. While we would like to prove that it is optimal, we cannot even overcome Challenge 1 which is generally much weaker (e.g., for the values $\delta = \epsilon = 0.01$).

Challenge 1 deals with the specific problem of finding a 3-way collision. Similar challenges can be formulated for other short-output cryptanalytic problems whose most time-efficient algorithm seems to require a large amount of space. The adaptation is performed by adjusting the distribution on inputs and the exponent 2/3 according to the specific problem. For example, for the problem of breaking double encryption we would consider an exponent of 1. In the next section, we propose a restricted model of computation which allows to bypass the challenge for the specific case of breaking double encryption.

6 A Time-Space Tradeoff Lower Bound for Post-Filtering Attacks on Double Encryption

Double encryption is one of the most fundamental constructions in symmetric-key cryptography. The classical meet-in-the-middle attack on the scheme (due to Merkle and Hellman [26]) gives the time-space tradeoff $T \cdot S = \tilde{O}(K^2)$ (where $(k_1, k_2) \in [K]^2$ is the key). This tradeoff was improved by van Oorschot and Wiener to $T^2 \cdot S = \tilde{O}(K^3)$ using the PCS algorithm [32]. In terms of lower bounds, the scheme is known to be secure up to $T = O(K)$ queries [1]. On the other hand, there are no known unconditional lower bounds that take into consideration space complexity for $S \ll T$. Indeed, in Section 5 we argued that proving such bounds may be very difficult.

In this section, we analyze the security of double encryption assuming that the space of the adversary is bounded. Our setting is similar to the one considered by Tessaro and Thiruvengadam in [31]. However, [31] reduced problem of breaking double encryption (i.e., distinguishing the scheme from a random permutation) to solving the element distinctness problem, and thus obtained a conditional result based on the current state-of-the-art for element distinctness algorithms. On the other hand, we obtain an unconditional security proof for a class of algorithms which is restricted, yet broad enough to capture the best known space-efficient attack algorithm (and its potential generalizations).

Let $E : [K] \times [N] \to [N]$ be a block cipher, which is a permutation on $[N]$ for each $k \in [K]$. The inverse block cipher is denoted by E^{-1}. Given block ciphers $E1, E2$, double encryption $DE : [K] \times [K] \times [N] \to [N]$ is defined as

$$DE_{k_1,k_2}(p) = E2_{k_2}(E1_{k_1}(p)),$$

for keys $(k_1, k_2) \in [K]^2$.

Let $BC_{K,N}$ be the set of all block ciphers with key space $[K]$ and block space $[N]$. Throughout this section, we assume for the sake of simplicity that $K = N$ and that $E1$ is independent of $E2$. It is not difficult to extend our results (with negligible loss in the bound) to the case of $K \neq N$ (as long as $K = O(N)$) and\or $E1 = E2$.

Recall from Section 1.1 that in the attack based on collision search, except for the main (p_1, c_1) plaintext-ciphertext pair, all other pairs are accessed only for post-filtering purposes. We now define a model which captures this attack and potentially additional post-filtering attacks on double-encryption (the model also captures the classical meet-in-the-middle attack [26]). Using this model, we prove that the time-space tradeoff obtained by the best know attack (which is based on PCS) is optimal for post-filtering algorithms.

We consider a post-filtering adversary $\mathcal{A}$ that attempts to distinguish between the real world (where ciphertexts are generated by a double encryption scheme) and an ideal world (where ciphertexts are generated at random). The adversary has access to the following functionalities:

1. Block ciphers $E1, E2 \in BC_{N,N}$ (chosen uniformly at random from the space of block ciphers), along with their inverses $E1^{-1}, E2^{-1}$.
2. In the real world, an arbitrary plaintext $p \in [N]$, along with $c = DE_{k_1,k_2}(p)$ for uniformly and independently chosen $(k_1, k_2) \in [N]^2$. In the ideal world, the adversary receives p and a uniformly chosen $c \in [N]$.
3. A post-filtering oracle $\mathcal{O} : [N]^2 \to \{0, 1\}$. In the real world, $\mathcal{O}_{(k_1,k_2)}(k_1', k_2') = 1$ if $(k_1', k_2') = (k_1, k_2)$ and $\mathcal{O}_{(k_1,k_2)}(k_1', k_2') = 0$ otherwise. In the ideal world, $\mathcal{O} = \mathcal{O}_\perp$ returns 0 on any input.

The access to the post-filtering oracle $\mathcal{O}$ is restricted, as it is only invoked on candidates (k_1', k_2') that satisfy $c = E2_{k_2'}(E1_{k_1'}(p))$. We thus assume that if $\mathcal{A}$ calls $\mathcal{O}$ with input (k_1', k_2') such that $c \neq E2_{k_2'}(E1_{k_1'}(p))$, the algorithm is terminated with failure.

The adversary issues T queries to $E1, E2$ and their inverses and has space of S bits. Finally, the adversary outputs a bit which represents a guess as to whether the interaction occurred in the real world, or in the ideal world.

Formally, we define the advantage of the adversary in the post-filtering double encryption (PFDE) game as

$$\begin{aligned}\mathrm{Adv}(\mathcal{A})^{\mathrm{PFDE}}_{DE[E1,E2]} = \\ |\Pr[E1, E2 \xleftarrow{\$} BC_{N,N}, (k_1, k_2) \xleftarrow{\$} [N]^2 : \\ \mathcal{A}^{E1,E1^{-1},E2,E2^{-1},\mathcal{O}_{(k_1,k_2)}}(p, c = DE_{k_1,k_2}(p)) = 1] - \\ \Pr[E1, E2 \xleftarrow{\$} BC_{N,N}, c \xleftarrow{\$} [N] : \mathcal{A}^{E1,E1^{-1},E2,E2^{-1},\mathcal{O}_\perp}(p, c) = 1]|.\end{aligned}$$

The main result of this section is given by the theorem below.

Theorem 3. *Let N, S, T be parameters such that $N \geq 3000, S \geq 5(\log^2 N + \log N)$. Any adversary $\mathcal{A}$ with space of S bits that makes at most T queries to $E1, E2$ and $E1^{-1}, E2^{-1}$ satisfies*

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{PFDE}}_{DE[E1,E2]} \leq \min\left(\frac{T^2}{N^2}, 288e \cdot \log N \cdot \frac{T\sqrt{S}}{N^{3/2}} + N^{-1/2}\right).$$

Hence, the advantage is $o(1)$ unless $T = \tilde{\Omega}\left(\frac{N^{3/2}}{S^{1/2}}\right)$, matching the best known attack.

6.1 Proof Overview

In order to prove Theorem 3, we first define the *restricted* post-filtering double-encryption game (RPFDE). The difference between this game and its unrestricted version above is that the adversary can only query $E1_k(p)$ and $E2_k^{-1}(c)$ for any choice of k, but cannot issue any other query. In Lemma 4, we show that despite the restriction on the adversary's queries in RPFSE, the distinguishing advantage remains the same as in PFSE. Hence it is sufficient to analyze RPFSE.

Next, we denote $f_1(k) = E1_k(p)$ and $f_2(k) = E2_k^{-1}(c)$, which syntactically transforms RPFSE to the notation used in Section 4 and allows to define the equivalent post-filtering collision search (PFCS) game. The goal is to show that in order to distinguish between the real and ideal worlds in PFCS (and RPFSE) with high probability, the adversary has to find $\Omega(N)$ collisions between f_1 and f_2 in the real world. Indeed, there are about N possible collisions between f_1 and f_2, but only one of them suggests the correct key and is accepted by the post-filtering oracle. Since the adversary is forced to find $\Omega(N)$ collisions, we can apply Theorem 2 to bound the success probability based on the adversary's time and space.

Applying Theorem 2 is not immediate since the assumption in this theorem is that f_1 and f_2 are independent, but in PFCS (and RPFSE) the functions are not independent, as they are known to collide for the correct choice of key. Hence, the application of Theorem 2 is made possible after an additional (hybrid argument) step that bounds the statistical distance between the dependent and independent distributions on (f_1, f_2).

Overall, the proof is somewhat more involved than one may expect. One reason for this is that we aim to prove security for parameter ranges of $T = \omega(N)$ (assuming $S = o(N)$), whereas standard security analysis of double encryption is only valid up to $T = N$. Consequently, some simple proof strategies that work up to $T = N$ are not good enough for our purposes.

Throughout the rest of this section, we denote $\alpha = \alpha(N) = 24e \log N$ (this expression appears in the time-space tradeoff formula of Theorem 2).

6.2 Restricted Post-Filtering Double Encryption

As noted above, the difference between PFDE and its restricted version is that in RPFDE the adversary can only query $E1_k(p)$ and $E2_k^{-1}(c)$ for any choice of k, but cannot issue any other query. We denote the advantage of the adversary in the restricted game as $\mathrm{Adv}(\mathcal{A})^{\mathrm{RPFDE}}_{DE[E1,E2]}$.

Theorem 3 follows from the two lemmas below. The first lemma shows that the restricted game does not hurt the distinguishing advantage of the adversary. The second lemma upper bounds the distinguishing advantage in RPFDE and its proof is given in Section 6.3.

Lemma 3. *Let N, S, T be parameters. If there exists an adversary $\mathcal{A}$ with space of S bits that makes at most T queries to $E1, E2$ and $E1^{-1}, E2^{-1}$ in the PFDE*

game, then there exists an adversary $\mathcal{A}'$ in the RPFDE game with space S and time T such that

$$\mathrm{Adv}(\mathcal{A}')^{\mathrm{RPFDE}}_{DE[E1,E2]} = \mathrm{Adv}(\mathcal{A})^{\mathrm{PFDE}}_{DE[E1,E2]}.$$

Lemma 4. *Let N, S, T be parameters such that $N \geq 3000, S \geq 5(\log^2 N + \log N)$. Then, any adversary $\mathcal{A}$ with space of S bits that makes at most T (restricted) queries to $E1$ and $E2^{-1}$ in the RPFDE game satisfies*

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{RPFDE}}_{DE[E1,E2]} \leq \frac{12\alpha \cdot T\sqrt{S}}{N\sqrt{N}} + N^{-1/2}.$$

Proof (of Theorem 3). First, $\mathrm{Adv}(\mathcal{A})^{\mathrm{PFDE}}_{DE[E1,E2]} \leq \frac{T^2}{N^2}$ by [1], which provides a general distinguishing advantage bound for double encryption that obviously holds here as well.

Moreover, by lemmas 3 and 4,

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{PFDE}}_{DE[E1,E2]} = \mathrm{Adv}(\mathcal{A}')^{\mathrm{RPFDE}}_{DE[E1,E2]} \leq \frac{12\alpha \cdot T\sqrt{S}}{N\sqrt{N}} + N^{-1/2}.$$

■

Proof (of Lemma 3). Given black-box access to adversary $\mathcal{A}$, we describe adversary $\mathcal{A}'$ that can only issue queries of the form $E1_k(p)$ and $E2_k^{-1}(c)$ for an arbitrary choice of k. In order to simulate answers to additional queries to $E1, E2, E1^{-1}$ and $E2^{-1}$, $\mathcal{A}'$ will utilize randomness that is independent of $E1, E2$ and used in order to construct block ciphers $E1', E2' : [N] \times [N] \to [N]$ that are chosen uniformly at random from $BC_{N,N}$, subject to the constraint that for each $k \in [N]$, $E1'_k(p) = E1_k(p)$ and $(E2'_k)^{-1}(c) = E2_k^{-1}(c)$.

The adversary $\mathcal{A}'$ runs $\mathcal{A}$ and answers every query to $E1$ or $E2$ (or their inverses) by issuing an identical query to $E1'$ or $E2'$ (or their inverses) and feeding the answer back to $\mathcal{A}$. Access to $\mathcal{O}$ remains identical. Finally, $\mathcal{A}'$ outputs the same value as $\mathcal{A}$.

We now describe how $E1', E2'$ are constructed. For each $k \in [N]$, the randomness of $\mathcal{A}'$ simply complements the constraint $E1'_k(p) = E1_k(p)$ to a random permutation (under this constraint), and similarly, complements the constraint $(E2'_k)^{-1}(c) = E2_k^{-1}(c)$ to a random permutation (under this constraint). Such randomness is independent of $E1, E2$, while a query to $E1', E2', (E1')^{-1}, (E2')^{-1}$ can be answered by querying $E1_k(p)$ (or $E2_k^{-1}(c)$) and the randomness.

It remains to analyze the complexity and advantage of $\mathcal{A}'$. We start by analyzing its advantage. First, note that for any k'_1, k'_2 such that $c = E2_{k'_2}(E1_{k'_1}(p))$, we have $c = E2'_{k'_2}(E1'_{k'_1}(p))$, hence the behaviour of $\mathcal{O}$ remains unchanged by the simulation (it is only invoked on legal inputs). Second, $\mathcal{A}'$ perfectly simulates the distribution of answers of $E1, E2, E1^{-1}, E2^{-1}$ in both the real and the ideal worlds. In other words, for every choice of $E1, E2$ in the real world, there is an equally likely choice of $E1' = E1, E2' = E2$ in the real world for which $\mathcal{A}'$ with access to $E1', E2'$ answers the same as $\mathcal{A}$ (and a similar statement holds in the ideal world). We conclude that $\mathrm{Adv}(\mathcal{A}')^{\mathrm{RPFDE}}_{DE[E1,E2]} = \mathrm{Adv}(\mathcal{A})^{\mathrm{PFDE}}_{DE[E1,E2]}$.

In terms of complexity, the block ciphers $E1', E2'$ are constructed such that every query to $E1', E2'$ (or their inverses) can be answered with at most one query to $E1_k(p)$ or $E2_k^{-1}(c)$ (for the same value of k). Since $\mathcal{A}$ makes at most T queries to $E1, E2, E1^{-1}, E2^{-1}$, then $\mathcal{A}'$ makes at most T such (restricted) queries. Furthermore, $\mathcal{A}'$ uses essentially the same space as $\mathcal{A}$ (in our model, the use of randomness is not counted towards the space nor the time complexity). ■

Remark 3. In the proof of Lemma 3, it may be tempting to implement $E1', E2' : [N]^2 \to [N]$ as independent block ciphers, and to query them for each query of $\mathcal{A}$ which is not to $E1_k(p)$ or $E2_k^{-1}(c)$. The problem with this implementation is that the answers that $\mathcal{A}$ receives for queries to $E1, E2$ (and their inverses) may no longer form a permutation for each $k \in [N]$, as they may contain a collision in the plaintext-ciphertext space for each k (due to the inconsistency between $E1$ and $E1'$ and between $E2$ and $E2'$). A single collision per $k \in [N]$ may not be a concern when $\mathcal{A}$ issues only $T \ll N$ queries, but in our case $T = \omega(N)$ (for $S = o(N)$) is possible.

6.3 Post-Filtering Collision Search

Towards proving Lemma 4, we first translate the cryptographic setting of double encryption to the more generic setting of Section 4 and relate these settings in Lemma 5 below. Lemma 4 then follows from Lemma 5 and Lemma 6 below (whose proof is given in Section 6.4) that bounds the adversary's advantage in the setting of Section 4.

Let $\mathcal{F} = \{f : [N] \to [N]\}$. We now define the post-filtering collision search (PFCS) game, where an algorithm $\mathcal{A}$ has access to functions $f_1, f_2 : [N] \to [N]$ and a post-filtering oracle $\mathcal{O} : [N]^2 \to \{0, 1\}$, initialized as follows:

1. In the real world, $(i_1, i_2) \in [N]^2$ is chosen uniformly at random. Then $f_1, f_2 : [N] \to [N]$ are chosen uniformly at random, subject to the constraint that $f_1(i_1) = f_2(i_2)$. We denote this distribution on (f_1, f_2, i_1, i_2) by $\mathcal{D}_2$. We define $\mathcal{O}_{(i_1,i_2)}(i'_1, i'_2) = 1$ if $(i'_1, i'_2) = (i_1, i_2)$ and $\mathcal{O}_{(i_1,i_2)}(i'_1, i'_2) = 0$ otherwise.
2. In the ideal world, $f_1, f_2 : [N] \to [N]$ are chosen uniformly at random and $\mathcal{O}_\perp$ returns 0 on any input.

As previously, access to the post-filtering oracle $\mathcal{O}$ is restricted, and it is only invoked on candidates (i'_1, i'_2) that satisfy $f_1(i'_1) = f_2(i'_2)$ (otherwise $\mathcal{A}$ is terminated). We define the advantage of the algorithm in the post-filtering collision search game as

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{PFCS}}_{f_1,f_2} =$$
$$|\Pr[(f_1, f_2, i_1, i_2) \leftarrow \mathcal{D}_2 : \mathcal{A}^{f_1,f_2,\mathcal{O}_{(i_1,i_2)}} = 1] - \Pr[f_1, f_2 \xleftarrow{\$} \mathcal{F} : \mathcal{A}^{f_1,f_2,\mathcal{O}_\perp} = 1|.$$

The PFCS game is merely a syntactical transformation of the RPFDE game, hence the following lemma is straightforward.

Lemma 5. *Let N, S, T be parameters. If there exists an adversary $\mathcal{A}$ with space of S bits that makes at most T queries to $g_1(k) = E1_k(p)$ and $g_2(k) = E2_k^{-1}(c)$ in the RPFCS game, then there exists an algorithm $\mathcal{A}'$ in the PFCS game with space S that makes at most T queries to f_1 and f_2 such that*

$$\mathrm{Adv}(\mathcal{A}')^{\mathrm{PFCS}}_{f_1,f_2} = \mathrm{Adv}(\mathcal{A})^{\mathrm{RPFDE}}_{DE[E1,E2]}.$$

Proof. Denoting $g_1(k) = E1_k(p)$ and $g_2(k) = E2_k^{-1}(c)$ as in the theorem, (g_1, g_2, k_1, k_2) in the real world is distributed according to $\mathcal{D}_2$, while g_1, g_2 in the ideal world are uniform and independent functions. Hence, given black-box access to an adversary $\mathcal{A}$ for RPFCS, an algorithm $\mathcal{A}'$ in PFCS with the desired properties can be constructed in a straightforward manner. ■

In the following, we will prove:

Lemma 6. *Let N, T, S be parameters such that $N \geq 3000, S \geq 5(\log^2 N + \log N)$. Then, any algorithm $\mathcal{A}$ for PFCS that queries f_1 and f_2 on T inputs and has space complexity of S bits satisfies*

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{PFCS}}_{f_1,f_2} \leq \frac{12\alpha \cdot T\sqrt{S}}{N\sqrt{N}} + N^{-1/2}.$$

Based on this lemma, we can prove Lemma 4.
Proof (of Lemma 4). By lemmas 5 and 6,

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{RPFDE}}_{DE[E1,E2]} = \mathrm{Adv}(\mathcal{A}')^{\mathrm{PFCS}}_{f_1,f_2} \leq \frac{12\alpha \cdot T\sqrt{S}}{N\sqrt{N}} + N^{-1/2}.$$

■

6.4 Bounding the Advantage in Post-Filtering Collision Search

It remains to prove Lemma 6. The proof is by a hybrid argument. We define *world 1* as an intermediate between the real and ideal worlds in PFCS. In world 1, algorithm $\mathcal{A}$ has access to f_1, f_2 and an oracle $\mathcal{O}$, initialized as follows:

1. The functions $f_1, f_2 : [N] \to [N]$ are chosen uniformly at random. Then, an index pair $(i_1, i_2) \in [N]^2$ is chosen uniformly at random from the collision set $\{(i'_1, i'_2) \mid f_1(i'_1) = f_2(i'_2)\}$ (if the collision set is empty, define $(i_1, i_2) = (0, 0)$). We denote this distribution on (f_1, f_2, i_1, i_2) by $\mathcal{D}_1$.
2. If the set $\{(i'_1, i'_2) \mid f_1(i'_1) = f_2(i'_2)\}$ is empty, then $\mathcal{O} = \mathcal{O}_\perp$ returns 0 on any input. If the collision set is non-empty, $\mathcal{O}_{(i_1,i_2)}(i'_1, i'_2) = 1$ if $(i'_1, i'_2) = (i_1, i_2)$ and $\mathcal{O}_{(i_1,i_2)}(i'_1, i'_2) = 0$ otherwise.

We define Game 1 as the problem of distinguishing the real world in PFCS from world 1, and Game 2 as the problem of distinguishing world 1 from the ideal world in PFCS. Correspondingly, we define

$$\begin{aligned}\mathrm{Adv}(\mathcal{A})^{\mathrm{G1}}_{f_1,f_2} = \\ |\Pr[(f_1, f_2, i_1, i_2) \leftarrow \mathcal{D}_2 : \mathcal{A}^{f_1,f_2,\mathcal{O}_{(i_1,i_2)}} = 1] - \\ \Pr[(f_1, f_2, i_1, i_2) \leftarrow \mathcal{D}_1 : \mathcal{A}^{f_1,f_2,\mathcal{O}_{(i_1,i_2)}} = 1]|,\end{aligned}$$

and

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{G2}}_{f_1,f_2} =$$

$$|\Pr[(f_1,f_2,i_1,i_2) \leftarrow \mathcal{D}_1 : \mathcal{A}^{f_1,f_2,\mathcal{O}_{(i_1,i_2)}} = 1] - \Pr[f_1,f_2 \xleftarrow{\$} \mathcal{F} : \mathcal{A}^{f_1,f_2,\mathcal{O}_\perp} = 1|.$$

We will prove the following two lemmas.

Lemma 7. *Any algorithm $\mathcal{A}$ in Game 1 satisfies*

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{G1}}_{f_1,f_2} \leq N^{-1/2} + 2e^{-N/120}.$$

Lemma 8. *Let N,T,S be parameters such that $N \geq 3000, S \geq 5(\log^2 N + \log N)$. Then, any algorithm $\mathcal{A}$ in Game 2 that makes T queries to f_1 and f_2 and has space of S bits satisfies*

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{G2}}_{f_1,f_2} \leq \frac{10\alpha \cdot T\sqrt{S}}{N\sqrt{N}}.$$

Proof (of Lemma 6). By a hybrid argument,

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{PFCS}}_{f_1,f_2} \leq \mathrm{Adv}(\mathcal{A})^{\mathrm{G1}}_{f_1,f_2} + \mathrm{Adv}(A)^{\mathrm{G2}}_{f_1,f_2} \leq$$

$$\frac{10\alpha \cdot T\sqrt{S}}{N\sqrt{N}} + N^{-1/2} + 2e^{-N/120} \leq \frac{12\alpha \cdot T\sqrt{S}}{N\sqrt{N}} + N^{-1/2},$$

where the penultimate inequality is due to lemmas 7 and 8, and the final inequality follows since $\alpha = 24e\log N$ and $N \geq 3000$. ■

It remains to prove lemmas 7 and 8. The proof of these lemmas requires an auxiliary lemma whose proof is given in Appendix B. We denote $Col(f_1,f_2) = |\{(i_1,i_2) \mid f_1(i_1) = f_2(i_2)\}|$, i.e., the size of the collision set. Lemma 9 provides concentration inequalities for $Col(f_1,f_2)$, when f_1,f_2 are independent random functions (which is the case when they are chosen according to $\mathcal{D}_1$).

Lemma 9. *Let $c > 0$ be any constant and suppose that f_1,f_2 are selected independently and uniformly at random from $\mathcal{F}$. Then,*

$$\Pr_{f_1,f_2}[|\, Col(f_1,f_2) - N \,| \geq c \cdot N^{1/2}] \leq c^{-2}.$$

Moreover,

$$\Pr_{f_1,f_2}[Col(f_1,f_2) < N/8] \leq 4e^{-N/120}.$$

Remark 4. It is possible to prove concentration inequalities for $Col(f_1,f_2)$ which are sharper than the ones of Lemma 9. However, Lemma 9 is sufficient for our purposes and is relatively easy to prove.

Proof (of Lemma 7). We have

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{G1}}_{f_1,f_2} \leq SD(\mathcal{D}_1,\mathcal{D}_2),$$

where $SD(\mathcal{D}_1, \mathcal{D}_2)$ is the statistical distance between $\mathcal{D}_1$ and $\mathcal{D}_2$. Hence, it suffices to prove that

$$SD(\mathcal{D}_1, \mathcal{D}_2) \leq N^{-1/2} + 2e^{-N/120}.$$

We denote by Λ the space

$$\{(f_1, f_2, i_1, i_2) \in \mathcal{F} \times \mathcal{F} \times [N] \times [N] \mid f_1(i_1) = f_2(i_2)\},$$

where $|\Lambda| = N^{2N+1}$. Recall that in order to sample according to $\mathcal{D}_2$, we first sample a uniform index pair (i_1, i_2) and then uniformly sample (f_1, f_2) under the restriction $f_1(i_1) = f_2(i_2)$. Hence, $\mathcal{D}_2$ is the uniform distribution over Λ, namely, for each $(f_1', f_2', i_1', i_2') \in \Lambda$,

$$\Pr_{(f_1,f_2,i_1,i_2)\leftarrow\mathcal{D}_2}[(f_1, f_2, i_1, i_2) = (f_1', f_2', i_1', i_2')] = 1/|\Lambda| = N^{-2N-1}.$$

On the other hand, in order to sample according to $\mathcal{D}_1$, we first sample (f_1, f_2) uniformly and then sample (i_1, i_2) from the collision set. Therefore, for each $(f_1', f_2', i_1', i_2') \in \Lambda$,

$$\Pr_{(f_1,f_2,i_1,i_2)\leftarrow\mathcal{D}_1}[(f_1, f_2, i_1, i_2) = (f_1', f_2', i_1', i_2')] =$$
$$\Pr[(f_1, f_2) = (f_1', f_2')] \cdot \Pr[(i_1, i_2) = (i_1', i_2') \mid (f_1, f_2) = (f_1', f_2')] =$$
$$N^{-2N} \cdot \frac{1}{Col(f_1', f_2')} = \frac{N}{Col(f_1', f_2') \cdot |\Lambda|},$$

whereas

$$\Pr_{(f_1,f_2,i_1,i_2)\leftarrow\mathcal{D}_1}[(f_1, f_2, i_1, i_2) \notin \Lambda] = \Pr[Col(f_1, f_2) = 0] \leq 4e^{-N/120},$$

by the second part of Lemma 9. Hence, treating the distributions $\mathcal{D}_1, \mathcal{D}_2$ as vectors over $\mathcal{F} \times \mathcal{F} \times [N] \times [N]$,

$$SD(\mathcal{D}_1, \mathcal{D}_2) =$$
$$1/2 \cdot \sum_{(f_1,f_2,i_1,i_2)\in\Lambda} |\mathcal{D}_1(f_1, f_2, i_1, i_2) - \mathcal{D}_2(f_1, f_2, i_1, i_2)| +$$
$$1/2 \cdot \sum_{(f_1,f_2,i_1,i_2)\notin\Lambda} |\mathcal{D}_1(f_1, f_2, i_1, i_2) - \mathcal{D}_2(f_1, f_2, i_1, i_2)| \leq$$
$$1/2 \cdot \sum_{(f_1,f_2,i_1,i_2)\in\Lambda} |\mathcal{D}_1(f_1, f_2, i_1, i_2) - \mathcal{D}_2(f_1, f_2, i_1, i_2)| + 2e^{-N/120}.$$

It remains to upper bound the first term above by $N^{-1/2}$. We have

$$1/2 \cdot \sum_{(f_1,f_2,i_1,i_2)\in\Lambda} |\mathcal{D}_1(f_1, f_2, i_1, i_2) - \mathcal{D}_2(f_1, f_2, i_1, i_2)| \leq$$
$$1/2 \cdot \frac{1}{|\Lambda|} \cdot \sum_{(f_1,f_2,i_1,i_2)\in\Lambda} \left| \frac{N}{Col(f_1, f_2)} - 1 \right| =$$

$$1/2 \cdot \frac{1}{|\Lambda|} \cdot \sum_{(f_1,f_2)} \left(\sum_{\{(i_1,i_2)|f_1(i_1)=f_2(i_2)\}} \left| \frac{N}{Col(f_1,f_2)} - 1 \right| \right) =$$

$$1/2 \cdot \frac{1}{|\Lambda|} \cdot \sum_{(f_1,f_2)} Col(f_1,f_2) \cdot \left| \frac{N}{Col(f_1,f_2)} - 1 \right| =$$

$$1/2 \cdot \frac{1}{|\Lambda|} \cdot \sum_{(f_1,f_2)} |N - Col(f_1,f_2)| =$$

$$1/2 \cdot N^{-1} \cdot \mathrm{E}_{f_1,f_2}[|Col(f_1,f_2) - N|] =$$

$$1/2 \cdot N^{-1} \cdot \sum_{i=0}^{\infty} \Pr_{f_1,f_2}[|Col(f_1,f_2) - N| \geq i] =$$

$$1/2 \cdot N^{-1} \cdot \left(\sum_{i=0}^{N^{1/2}-1} \Pr_{f_1,f_2}[|Col(f_1,f_2) - N| \geq i] + \sum_{i=N^{1/2}}^{\infty} \Pr_{f_1,f_2}[|Col(f_1,f_2) - N| \geq i] \right) \leq$$

$$1/2 \cdot N^{-1} \cdot \left(N^{1/2} + \sum_{i=N^{1/2}}^{\infty} \Pr_{f_1,f_2}\left[|Col(f_1,f_2) - N| \geq (i \cdot N^{-1/2}) \cdot N^{1/2}\right] \right) \leq$$

$$1/2 \cdot N^{-1} \left(N^{1/2} + \sum_{i=N^{1/2}}^{\infty} (i \cdot N^{-1/2})^{-2} \right) = 1/2 \cdot \left(N^{-1/2} + \sum_{i=N^{1/2}}^{\infty} i^{-2} \right) \leq N^{-1/2},$$

where the penultimate inequality is by the first part of Lemma 9. This completes the proof. ■

Remark 5. In $\mathcal{D}_2$, the dependency of f_1 and f_2 is only due to the index pair (i_1, i_2). Such a dependency is unnoticeable to an algorithm $\mathcal{A}$ as long as it does not query both i_1 and i_2, which occurs with probability of at most $\frac{T^2}{N^2}$. Hence, if we were interested in bounding the advantage of $\mathcal{A}$ only up to $T = N$, we could easily replace the proof of Lemma 7 by a simpler proof. However, it is not clear how to obtain such a simple proof that gives a meaningful bound for $T = \omega(N)$ (when $S = o(N)$).

Proof (of Lemma 8). Denote by $\mathcal{E}$ the event that $\mathcal{O}$ is invoked with (i_1, i_2) (and answers 1) in world 1. Note that $\mathrm{Adv}(\mathcal{A})^{\mathrm{G2}}_{f_1,f_2} \leq \Pr[\mathcal{E}]$, as conditioned on $\neg\mathcal{E}$ in world 1, both worlds are identical and the advantage is 0.

We focus on world 1. For $C \geq 0$, denote by $\mathcal{E}_C$ the event that $\mathcal{A}^{f_1,f_2,\mathcal{O}}$ calls the oracle $\mathcal{O}$ with at most C distinct pairs (i'_1, i'_2) such that $f_1(i'_1) = f_2(i'_2)$. According to the distribution $\mathcal{D}_1$, the probability that any pair (i'_1, i'_2) satisfies $(i'_1, i'_2) = (i_1, i_2)$ is $1/Col(f_1, f_2)$. Hence, for any positive value of $Col(f_1, f_2)$ and $0 \leq C \leq Col(f_1, f_2)$,

$$\Pr[\mathcal{E} \mid \mathcal{E}_C] \leq \frac{C}{Col(f_1,f_2)}.$$

By the above inequality and the second part of Lemma 9,

$$\Pr[\mathcal{E} \mid \mathcal{E}_C] \leq$$
$$\Pr[\mathcal{E} \mid \mathcal{E}_C \wedge Col(f_1, f_2) \geq N/8] + \Pr[Col(f_1, f_2) < N/8] \leq \frac{8}{N} \cdot C + 4e^{-N/120}. \tag{3}$$

Define $\hat{C} = \frac{\alpha \cdot T\sqrt{S}}{\sqrt{N}}$. We have

$$\mathrm{Adv}(\mathcal{A})^{\mathrm{G2}}_{f_1,f_2} \leq \Pr_{(f_1,f_2,i_1,i_2)\leftarrow\mathcal{D}_1}[\mathcal{E}] \leq \Pr\left[\mathcal{E} \mid \mathcal{E}_{\hat{C}}\right] + \Pr\left[\neg\mathcal{E}_{\hat{C}}\right] \leq$$
$$\frac{8}{N} \cdot \frac{\alpha \cdot T\sqrt{S}}{\sqrt{N}} + 4e^{-N/120} + N^{-2} \leq \frac{10\alpha \cdot T\sqrt{S}}{N\sqrt{N}},$$

where the penultimate inequality is by (3) and Theorem 2, and the final inequality follows since $\alpha = 24e \log N$ and $N \geq 3000$. ■

7 Conclusions and Future Work

In this paper we proved that the well-known time-space tradeoff $T^2 \cdot S = \tilde{O}(C^2 \cdot N)$ for the collision search problem is optimal using the framework of Borodin and Cook. We further proved that the best known time-space tradeoff attack on double encryption is optimal among post-filtering algorithms.

In the future it would be interesting to find more problems in cryptography for which time-space tradeoff lower bounds can be proved by the method of Borodin and Cook. Another research direction is to extend the post-filtering model and prove time-space tradeoff lower bounds for additional (short-output) cryptanalytic problems under reasonable restrictions.

Acknowledgements: The author was supported by the Israeli Science Foundation through grant No. 573/16 and by the European Research Council under the ERC starting grant agreement No. 757731 (LightCrypt).

References

1. W. Aiello, M. Bellare, G. D. Crescenzo, and R. Venkatesan. Security Amplification by Composition: The Case of Doubly-Iterated, Ideal Ciphers. In H. Krawczyk, editor, *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*, pages 390–407. Springer, 1998.
2. N. Alon, Y. Matias, and M. Szegedy. The Space Complexity of Approximating the Frequency Moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.
3. J. Alwen, B. Chen, K. Pietrzak, L. Reyzin, and S. Tessaro. Scrypt Is Maximally Memory-Hard. In J. Coron and J. B. Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 33–62, 2017.

4. J. Alwen and V. Serbinenko. High Parallel Complexity Graphs and Memory-Hard Functions. In R. A. Servedio and R. Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 595–603. ACM, 2015.
5. B. Auerbach, D. Cash, M. Fersch, and E. Kiltz. Memory-Tight Reductions. In J. Katz and H. Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 101–132. Springer, 2017.
6. E. Barkan, E. Biham, and A. Shamir. Rigorous Bounds on Cryptanalytic Time/Memory Tradeoffs. In C. Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 1–21. Springer, 2006.
7. P. Beame. A general sequential time-space tradeoff for finding unique elements. *SIAM J. Comput.*, 20(2):270–277, 1991.
8. P. Beame, R. Clifford, and W. Machmouchi. Element Distinctness, Frequency Moments, and Sliding Windows. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 290–299. IEEE Computer Society, 2013.
9. P. Beame, M. E. Saks, X. Sun, and E. Vee. Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003.
10. A. Borodin. Time Space Tradeoffs (Getting Closer to the Barrier?). In K. Ng, P. Raghavan, N. V. Balasubramanian, and F. Y. L. Chin, editors, *Algorithms and Computation, 4th International Symposium, ISAAC '93, Hong Kong, December 15-17, 1993, Proceedings*, volume 762 of *Lecture Notes in Computer Science*, pages 209–220. Springer, 1993.
11. A. Borodin and S. A. Cook. A Time-Space Tradeoff for Sorting on a General Sequential Model of Computation. *SIAM J. Comput.*, 11(2):287–297, 1982.
12. A. Borodin, M. J. Fischer, D. G. Kirkpatrick, N. A. Lynch, and M. Tompa. A Time-Space Tradeoff for Sorting on Non-Oblivious Machines. *J. Comput. Syst. Sci.*, 22(3):351–364, 1981.
13. C. Cachin and U. M. Maurer. Unconditional Security Against Memory-Bounded Adversaries. In B. S. K. Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 292–306. Springer, 1997.
14. A. Chakrabarti and Y. Chen. Time-Space Tradeoffs for the Memory Game. *CoRR*, abs/1712.01330, 2017.
15. A. Cobham. The Recognition Problem for the Set of Perfect Squares. In *7th Annual Symposium on Switching and Automata Theory, Berkeley, California, USA, October 23-25, 1966*, pages 78–87. IEEE Computer Society, 1966.
16. C. Delaplace, A. Esser, and A. May. Improved low-memory subset sum and LPN algorithms via multiple collisions. *IACR Cryptology ePrint Archive*, 2019:804, 2019.
17. I. Dinur, O. Dunkelman, N. Keller, and A. Shamir. Efficient Dissection of Composite Problems, with Applications to Cryptanalysis, Knapsacks, and Combinatorial Search Problems. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 719–740. Springer, 2012.
18. D. P. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

19. C. Dwork, M. Naor, and H. Wee. Pebbling and Proofs of Work. In V. Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 37–54. Springer, 2005.
20. L. Fortnow, R. J. Lipton, D. van Melkebeek, and A. Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):835–865, 2005.
21. M. E. Hellman. A cryptanalytic time-memory trade-off. *IEEE Transactions on Information Theory*, 26(4):401–406, 1980.
22. J. Jaeger and S. Tessaro. Tight Time-Memory Trade-Offs for Symmetric Encryption. In Y. Ishai and V. Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 467–497. Springer, 2019.
23. A. Joux and S. Lucks. Improved Generic Algorithms for 3-Collisions. In M. Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, volume 5912 of *Lecture Notes in Computer Science*, pages 347–363. Springer, 2009.
24. D. E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Addison-Wesley, 1969.
25. Y. Mansour, N. Nisan, and P. Tiwari. The Computational Complexity of Universal Hashing. *Theor. Comput. Sci.*, 107(1):121–133, 1993.
26. R. C. Merkle and M. E. Hellman. On the Security of Multiple Encryption. *Commun. ACM*, 24(7):465–467, 1981.
27. N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.
28. P. Oechslin. Making a Faster Cryptanalytic Time-Memory Trade-Off. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, volume 2729 of *Lecture Notes in Computer Science*, pages 617–630. Springer, 2003.
29. W. J. Paul, R. E. Tarjan, and J. R. Celoni. Space Bounds for a Game on Graphs. *Mathematical Systems Theory*, 10:239–251, 1977.
30. R. Raz. Fast Learning Requires Good Memory: A Time-Space Lower Bound for Parity Learning. *J. ACM*, 66(1):3:1–3:18, 2019.
31. S. Tessaro and A. Thiruvengadam. Provable Time-Memory Trade-Offs: Symmetric Cryptography Against Memory-Bounded Adversaries. In A. Beimel and S. Dziembowski, editors, *Theory of Cryptography - 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part I*, volume 11239 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2018.
32. P. C. van Oorschot and M. J. Wiener. Parallel Collision Search with Cryptanalytic Applications. *J. Cryptology*, 12(1):1–28, 1999.
33. D. A. Wagner. A Generalized Birthday Problem. In M. Yung, editor, *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002.
34. A. C. Yao. Probabilistic Computations: Toward a Unified Measure of Complexity (Extended Abstract). In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 222–227. IEEE Computer Society, 1977.

35. A. C. Yao. Near-Optimal Time-Space Tradeoff for Element Distinctness. *SIAM J. Comput.*, 23(5):966–975, 1994.
36. Y. Yesha. Time-Space Tradeoffs for Matrix Multiplication and the Discrete Fourier Transform on any General Sequential Random-Access Computer. *J. Comput. Syst. Sci.*, 29(2):183–197, 1984.

A The Parallel Collision Search Algorithm [32]

In this section, we briefly summarize the PCS algorithm for computing C colliding pairs in a random function $f : [N] \to [N]$. For more details, refer to [32]. Given $\tilde{O}(S)$ bits of memory, PCS builds a chain structure containing S chains, where a chain starts at an arbitrary point $x_0 \in [N]$ and computed iteratively as $x_{i+1} = f(x_i)$. Each chain is terminated after about $\sqrt{N/S}$ evaluations, hence the structure contains a total of about $S \cdot \sqrt{N/S} = \sqrt{N \cdot S}$ distinct points. As the chains are of length $\sqrt{N/S}$, each chain collides with a different chain in the structure with constant probability according to the birthday paradox, since the number of relevant pairs of points is $\sqrt{N/S} \cdot \sqrt{N \cdot S} = N$. Therefore, the structure contains an expected number of about S colliding pairs.

The collisions can be recovered efficiently by defining a set of $\sqrt{N \cdot S}$ distinguished points according to an easily verifiable condition on the points $x_i \in [N]$. Each chain in the structure is terminated at a distinguished point (and hence its expected length is $N/\sqrt{N \cdot S} = \sqrt{N/S}$ as required). The PCS algorithm stores the distinguished points sorted in memory and collisions between chains are detected at their distinguished points. The actual collisions in f are recovered by recomputing the colliding chains.

In total, PCS finds $C = \Theta(S)$ distinct colliding pairs in f using space of $\tilde{O}(S)$ bits and time complexity $T = \tilde{O}(\sqrt{N \cdot S})$.

When $C > S$ collisions are required, the algorithm is repeated $O(C/S)$ times. In order to (heuristically) eliminate the dependency between the different executions, in repetition i we run PCS on the function $f_i = \pi_i \circ f$, where $\pi_i : [N] \to [N]$ is some simple permutation. Note that a collision in f_i gives a collision in f. Altogether, PCS finds C distinct colliding pairs in f using space of $\tilde{O}(S)$ bits and time complexity $T = \tilde{O}(C/S \cdot \sqrt{N \cdot S}) = \tilde{O}(C \cdot \sqrt{N/S})$, which gives the time-space tradeoff curve $T^2 \cdot S = \tilde{O}(C^2 \cdot N)$.

B Proof of Lemma 9

Proof (of Lemma 9). We begin by proving the first part of the lemma. For every $(i_1, i_2) \in [N]^2$ define an indicator random variable $C_{i_1 i_2}$ that is equal to 1 if $f_1(i_1) = f_2(i_2)$. We have

$$\begin{aligned}
&\mathrm{E}[C_{i_1 i_2}] = \Pr[C_{i_1 i_2} = 1] = N^{-1}, \text{ and} \\
&\mathrm{Var}[C_{i_1 i_2}] = \mathrm{E}[(C_{i_1 i_2})^2] - (\mathrm{E}[C_{i_1 i_2}])^2 = N^{-1} - N^{-2} < N^{-1}.
\end{aligned}$$

Hence,

$$\mathrm{E}[Col(f_1, f_2)] = \mathrm{E}\left[\sum_{(i_1,i_2)\in[N]^2} C_{i_1 i_2}\right] = \sum_{(i_1,i_2)\in[N]^2} \mathrm{E}[C_{i_1 i_2}] = N^2 \cdot N^{-1} = N.$$

Since the random variables $\{C_{i_1 i_2}\}$ are pairwise independent,

$$\mathrm{Var}[Col(f_1, f_2)] = \mathrm{Var}\left[\sum_{(i_1,i_2)\in[N]^2} C_{i_1 i_2}\right] = \sum_{(i_1,i_2)\in[N]^2} \mathrm{Var}[C_{i_1 i_2}] < N^2 \cdot N^{-1} = N.$$

For a parameter $c > 0$, Chebyshev's inequality gives

$$\Pr\left[|Col(f_1, f_2) - \mathrm{E}[Col(f_1, f_2)]| \geq c \cdot \sqrt{\mathrm{Var}[Col(f_1, f_2)]}\right] \leq c^{-2}.$$

Therefore, we obtain

$$\Pr[|Col(f_1, f_2) - N| \geq c \cdot N^{1/2}] \leq c^{-2},$$

as required.

For the second part of the lemma, we view the process of sampling f_1 (and f_2) as a classical Balls-and-Bins problem, where we throw N balls into N bins uniformly at random, and ball i falls into bin $f_1(i)$. Denote by Z_1 the number of empty bins induced by f_1 (i.e., the number of points $x \in [N]$ with no preimage under f_1) and by Z_2 the number of empty bins induced by f_2. Hence, the number of non-empty bins (image points) induced by f_1 and f_2 are $N - Z_1$ and $N - Z_2$, respectively. The number of colliding pairs between f_1, f_2 is at least the size of the intersection of the non-empty bins, which is at least $(N-Z_1)+(N-Z_2)-N = N - Z_1 - Z_2$.

Hence, if $Col(f_1, f_2) < N/8$, then $N - Z_1 - Z_2 < N/8$, which implies that $Z_1 + Z_2 > 7N/8$. Therefore, either $Z_1 > 7N/16$, or $Z_2 > 7N/16$. By [18, p.75], we have

$$\Pr[|Z_1 - \mathrm{E}[Z_1]| > t] \leq 2e^{-2t^2/N},$$

and the same holds for Z_2.

The probability that any particular bin is empty is $(1 - N^{-1})^N \leq 1/e$, hence $\mathrm{E}[Z_1] \leq N/e$. Therefore,

$$\Pr[Z_1 > 7N/16] = \Pr[Z_1 - N/e > 7N/16 - N/e] \leq \Pr[Z_1 - \mathrm{E}[Z_1] > N/15] \leq \Pr[|Z_1 - \mathrm{E}[Z_1]| > N/15] \leq 2e^{-N/120}.$$

Finally,

$$\Pr[Col(f_1, f_2) < N/8] \leq \Pr[Z_1 > 7N/16] + \Pr[Z_2 > 7N/16] \leq 4e^{-N/120}.$$

■