Gate Elimination for Linear Functions and New Feebly Secure Constructions

Alex Davydow[1] and Sergey I. Nikolenko[2]

[1] St. Petersburg Academic University, ul. Khlopina, 8, korp. 3, St. Petersburg, Russia, adavydow@yandex.ru
[2] Steklov Mathematical Institute, nab. r. Fontanka, 27, St. Petersburg, Russia, sergey@logic.pdmi.ras.ru

Abstract. We consider gate elimination for linear functions and show two general forms of gate elimination that yield novel corollaries. Using these corollaries, we construct a new linear feebly secure trapdoor function that has order of security $\frac{5}{4}$ which exceeds the previous record for linear constructions. We also give detailed proofs for nonconstructive circuit complexity bounds on linear functions.

Keywords: circuit complexity, gate elimination, feebly secure cryptography, feebly one-way functions

1 Introduction

Modern cryptography has virtually no provably secure constructions. Starting from the first Diffie–Hellman key agreement protocol [5] and the first public key cryptosystem RSA [21], not a single public key cryptographic protocol has been proven secure (however, there exist secure secret key protocols, e.g., the one-time pad scheme [23, 25]). Naturally, an unconditional proof of security would be indeed hard to find, since it would necessarily imply that P $\neq$ NP. But the situation is worse: there are also no conditional proofs that might establish a connection between natural structural assumptions (like P$\neq$NP or BPP$\neq$NP) and cryptographic security. Recent developments in lattice-based cryptosystems relate cryptographic security with worst-case complexity, but they deal with problems unlikely to be NP-complete [1, 6, 19, 20].

There are known complete cryptographic constructions, both one-way functions [14,15] and public key cryptosystems [8,9]. However, they also do not let us relate cryptographic security to key assumptions of classical complexity theory. Moreover, the asymptotic nature of these completeness results does not let us say anything about how hard it is to break a given cryptographic protocol for keys of a certain fixed length, which is, in fact, what we all want as privacy-aware customers. Problems that have been studied extensively in relation to cryptography (factoring and discrete logarithm) do seem to scale well, but there are no lower bounds and little hope for such anytime soon, so the point is moot. We can only say that complexity of the algorithms that we have devised ourselves

scales well, so ultimately we fall back on the "many smart people have thought about it" argument. There are other dangers on the way of asymptotic complexity, too [13]. Ultimately, we do not care whether a protocol can or cannot be broken in the limit; we would be very happy if breaking this specific version of the protocol required constant time, but the constant was larger than the size of the known Universe.

However, modern cryptography is still very far away from *provably* secure constructions. At present, we can prove security neither in this "hard" sense nor in the sense of classical cryptographic definitions [7]. Nevertheless, while we are unable to prove a superpolynomial gap between the complexities of honest parties and adversaries, we are able to prove *some* gap. In 1992, Alain Hiltgen [10] presented a series of linear functions that are about *twice* ($2 - \epsilon$ times for an arbitrarily small ϵ) harder to invert than to compute. His example consists of linear functions over $\mathbb{F}_2$ with matrices that have few non-zero entries (ones) while inverse matrices have many ones. The complexity gap follows by a simple argument of Lamagna and Savage [16, 22]: every bit of the output depends nonidly on many variables and all these bits correspond to different functions, hence a lower bound on the complexity of computing them all together. The model of computation here is the most general one, namely the number of gates in a Boolean circuit that uses arbitrary binary Boolean gates. Little more could be expected for this model at present. For example, the best known lower bound for general circuit complexity of a specific Boolean function is $3n - o(n)$ [3, 26]. In his thesis, Hiltgen also presented a feebly secure function that is exactly twice harder to invert than to compute, this time a nonlinear one [11].

Lately, in the works of Hirsch, Nikolenko, and Melanich new cryptographic constructions with the same properties were constructed [12, 17]. In [12], a linear feebly secure trapdoor function with order of security $\frac{25}{22}$ was constructed, and in [17], a nonlinear feebly secure trapdoor function with order of security $\frac{7}{5}$. In this paper, we continue that line of work. In Section 2, we give basic definitions.

Virtually all bounds in general circuit complexity have been proven with *gate elimination*. This paper deals with gate elimination for linear functions; in Section 3, we distill gate elimination to two basic ideas, formulate it in a general form, note important corollaries, and discuss its limitations. Our discussion in Section 3 generalizes the methods of [12], and this understanding lets us find a better construction of a linear feebly secure trapdoor function that has order of security $\frac{5}{4} - \epsilon$ for any $\epsilon > 0$, as shown in Section 4. In Section 5, we compare what we have found with nonconstructive upper and lower bounds on general circuit complexity of linear functions. Section 6 concludes the paper.

2 Preliminaries

We denote by $\mathbb{B}_{n,m}$ the set of all 2^{m2^n} total functions $f : \mathbb{B}^n \to \mathbb{B}^m$, where $\mathbb{B} = \{0, 1\}$ is the field with two elements. A *circuit* is a directed acyclic labeled graph with vertices of two kinds: vertices of indegree 0 (vertices that no edges enter) labeled by one of the variables $x_1, \ldots, x_n$, and vertices labeled by a binary

Boolean function $f \in \mathbb{B}_{2,1}$; this model of computation is known as *general circuit complexity*. Vertices of the first kind are called *inputs* or input variables; vertices of the second kind, *gates*. The *size* of a circuit C, $\mathrm{size}(C)$, is the number of gates in it. We assume that each gate in this circuit depends of both inputs, i.e., there are no gates marked by constants and unary functions Id and $\neg$. To safely remove such gates without loss of generality, we assume that the output of each gate can be both the value of corresponding function and its negation. For every injective function of n variables $f_n \in \mathbb{B}_{n,m}$ we define its *measure of one-wayness* $M_F(f_n) = \frac{C(f_n^{-1})}{C(f_n)}$. Hiltgen's work was to find sequences of functions $f = \{f_n\}_{n=1}^{\infty}$ with a large asymptotic constant $\liminf_{n\to\infty} M_F(f_n)$, which is called the *order of one-wayness* of f.

There is a well-known definition in cryptography for a family of trapdoor functions [7]. However, we have a more detailed definition: since we are interested in constants here, we must pay attention to all the details.

Definition 1. *Fix functions* $\mathrm{pi}, \mathrm{ti}, m, c : \mathbb{N} \to \mathbb{N}$. *A* feebly trapdoor candidate *is a sequence of triples of circuits* $\mathcal{C} = \{(\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n)\}_{n=1}^{\infty}$ *where:*

- $\{\mathrm{Key}_n\}_{n=1}^{\infty}$ *is a family of sampling circuits* $\mathrm{Key}_n : \mathbb{B}^n \to \mathbb{B}^{\mathrm{pi}(n)} \times \mathbb{B}^{\mathrm{ti}(n)}$,
- $\{\mathrm{Eval}_n\}_{n=1}^{\infty}$ *is a family of evaluation circuits* $\mathrm{Eval}_n : \mathbb{B}^{\mathrm{pi}(n)} \times \mathbb{B}^{m(n)} \to \mathbb{B}^{c(n)}$, *and*
- $\{\mathrm{Inv}_n\}_{n=1}^{\infty}$ *is a family of inversion circuits* $\mathrm{Inv}_n : \mathbb{B}^{\mathrm{ti}(n)} \times \mathbb{B}^{c(n)} \to \mathbb{B}^{m(n)}$

such that for every security parameter n*, every seed* $s \in \mathbb{B}^n$*, and every input* $\mathrm{m} \in \mathbb{B}^{m(n)}$

$$\mathrm{Inv}_n(\mathrm{Key}_{n,2}(s), \mathrm{Eval}_n(\mathrm{Key}_{n,1}(s), \mathrm{m})) = \mathrm{m},$$

where $\mathrm{Key}_{n,1}(s)$ *and* $\mathrm{Key}_{n,2}(s)$ *are the first* $\mathrm{pi}(n)$ *bits ("public information") and the last* $\mathrm{ti}(n)$ *bits ("trapdoor information") of* $\mathrm{Key}_n(s)$*, respectively.*

Informally speaking, n is the security parameter (the length of the random seed), $m(n)$ is the length of the input to the function, $c(n)$ is the length of the function's output, and $\mathrm{pi}(n)$ and $\mathrm{ti}(n)$ are lengths of the public and trapdoor information, respectively. We call these functions "candidates" because Definition 1 does not imply any security, it merely sets up the dimensions and provides correct inversion. In our constructions, $m(n) = c(n)$ and $\mathrm{pi}(n) = \mathrm{ti}(n)$.

To find how secure a function is, we introduce the notion of a break. Informally, an adversary should invert the function without knowing the trapdoor information. We introduce break as inversion with probability greater than a certain constant r (we will usually set r to equal $\frac{1}{2}$, $\frac{3}{4}$, or $\frac{7}{8}$). We denote by $C_\alpha(f)$ the minimal size of a circuit that correctly computes a function $f \in \mathcal{B}_{n,m}$ on more than fraction α of its inputs (of length n). Obviously, $C_\alpha(f) \le C_\beta(f)$ for all $\alpha \le \beta$, and $C(f) = C_1(f)$.

Definition 2. *A circuit* N breaks *a feebly trapdoor candidate* $\mathcal{C} = \{\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n\}$ *on seed length* n *with probability* r *if, for uniformly chosen seeds* $s \in \mathbb{B}^n$ *and inputs* $m \in \mathbb{B}^{m(n)}$*,*

$$\Pr_{(s,m)\in U}\left[N(\mathrm{Key}_{n,1}(s), \mathrm{Eval}_n(\mathrm{Key}_{n,1}(s), m)) = m\right] > r.$$

Definition 3. *A feebly trapdoor candidate* $\mathcal{C} = \{\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n\}$ *has* order of security k *with level* α *if for every sequence of circuits* $\{N_n\}_{n=1}^{\infty}$ *that break* f *on every input length* n *with probability* α*,*

$$\lim_{n\to\infty} \inf \min \left\{ \frac{C(N_n)}{C(\mathrm{Key}_n)}, \frac{C(N_n)}{C(\mathrm{Eval}_n)}, \frac{C(N_n)}{C(\mathrm{Inv}_n)} \right\} \geq k.$$

In other words,

$$\lim_{n\to\infty} \inf \min \left\{ \frac{C_{3/4}(f_{\mathrm{pi}(n)+c(n)})}{C(\mathrm{Key}_n)}, \frac{C_{3/4}(f_{\mathrm{pi}(n)+c(n)})}{C(\mathrm{Eval}_n)}, \frac{C_{3/4}(f_{\mathrm{pi}(n)+c(n)})}{C(\mathrm{Inv}_n)} \right\} \geq k,$$

where the function $f_{\mathrm{pi}(n)+c(n)}$ *maps* $\big(\mathrm{Key}_{n,1}(s), \mathrm{Eval}_n(\mathrm{Key}_{n,1}(s), m)\big) \mapsto m$.

We list a few simple examples. If there is no secret key at all ($\mathrm{ti}(n) = 0$), each feebly trapdoor candidate $\{(\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n)\}_{n=1}^{\infty}$ has order of security 1, since the sequence of circuits $\{\mathrm{Inv}_n\}_{n=1}^{\infty}$ successfully inverts it. If a sequence $\{(\mathrm{Key}_n, \mathrm{Eval}_n, \mathrm{Inv}_n)\}_{n=1}^{\infty}$ implements a trapdoor function in the usual cryptographic sense, then $k = \infty$. Moreover, $k = \infty$ even if the bounds on adversary size are just a bit more than linear, e.g., if the adversary requires $O(n \log n)$ gates. Our definitions are not designed to distinguish between these (very different) cases, because, unfortunately, any nonlinear lower bound on general circuit complexity appears very far away from our current state of knowledge.

Let us also note explicitly that we are talking about *one-time* security. An adversary can amortize his circuit complexity on inverting a feebly trapdoor candidate for the second time for the same seed, for example, by computing the trapdoor information and successfully reusing it. Thus, in this setting one has to pick a new seed for every input.

3 Gate Elimination for Linear Functions

Gate elimination is virtually the only method we have to prove lower bounds in general circuit complexity; so far, it has been used for every single lower bound [3,18,24,26]. The basic idea of this method is to use the following inductive argument. Consider a function f and a circuit of minimal size C that computes it. Now substitute some value c for some variable x thus obtaining a circuit for the function $f\,|_{x=c}$. The original circuit C can now be simplified, because the gates that had this variable as inputs become either unary (recollect that the negation can be embedded into subsequent gates) or constant (in this case we can even proceed to eliminating subsequent gates). After figuring out how many gates one can eliminate on every step, one proceeds by induction as long as it is possible to find a suitable variable that eliminates enough gates. Evidently, the number of eliminated gates is a lower bound on the complexity of f.

Usually, the important case here is when a gate is nonlinear, such as an AND or an OR gate. In that case, it is always possible to choose a value for an input of such a gate so that this gate becomes a constant and, therefore, its immediate descendants can also be eliminated. However, in this paper we deal with gate

elimination for *linear* functions. We do not know how to prove that one cannot, in general, produce a smaller circuit for a linear function with nonlinear gates, but it is evident that we cannot assume any gates to be nonlinear in this setting. Thus, gate elimination distills to two very simple ideas. Idea 1 is trivial and has been noted many times before, while Idea 2 will allow us to devise better feebly secure constructions in Section 4.

Since we are dealing with linear functions, we will, for convenience, state our results in terms of matrices over $\mathbb{F}_2$; the circuit complexity of a matrix $C_\alpha(A)$ is the circuit complexity of the corresponding linear function. By A_{-i} we denote the matrix A without its i^{th} column; note that if A corresponds to f then A_{-i} corresponds to $f\mid_{x_i=0}$. If a matrix A has a zero column A_i, it means that the corresponding function does not depend on the input x_i; in what follows, we will always assume that functions depend nontrivially on all their inputs and thus the matrices do not have zero columns; we call such matrices *nontrivial*. Note that if A is a submatrix of B then $C_\alpha(A) \le C_\alpha(B)$ for all $\alpha \in [0,1]$.

Idea 1. Suppose that for n steps, there is at least one gate to eliminate. Then $C(f) \ge n$.

Theorem 1. *Fix a real number $\alpha \in [0,1]$. Suppose that $\mathcal{P} = \{P_n\}_{n=1}^{\infty}$ is a series of predicates defined on matrices over $\mathbb{F}_2$ with the following properties:*

- *if $P_1(A)$ holds then $C_\alpha(A) \ge 1$;*
- *if $P_n(A)$ holds then $P_m(A)$ holds for every $1 \le m \le n$;*
- *if $P_n(A)$ holds then, for every index i, $P_{n-1}(A_{-i})$ holds.*

Then, for every matrix A with $\ge n+1$ columns, if $P_n(A)$ holds then $C_\alpha(A) \ge n$.

Proof. The proof is a straightforward induction on the index of P_i; the first property of $\mathcal{P}$ provides the base, and other properties take care of the induction step. For the induction step, consider the first gate of an optimal circuit C implementing A. By the monotonicity property of $\mathcal{P}$ and the induction base, the circuit is nontrivial, so there is a first gate. Consider a variable x_i entering that gate. Note that if C computes f on fraction α of its inputs then for some c, $C\mid_{x_i=c}$ computes $f\mid_{x_i=c}$ on fraction α of its inputs. If we substitute this value into this variable, we get a circuit $C\mid_{x_i=c}$ that has at most $\text{size}(C)-1$ gates and implements A_{-i} on at least α fraction of inputs. □

The theorem is absolutely trivial; however, it has so far been the only instrument available for gate elimination in linear functions. In fact, the only instrument has been an even simpler proposition that dates back to mid-1970s.

Proposition 1 ([16, 22]; [10, Theorems 3 and 4]; [12, Proposition 1]).

1. *Suppose that $f : \mathbb{B}^n \to \mathbb{B}$ depends non-idly on each of its n variables, that is, for every i there exist values $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in \mathbb{B}$ such that $f(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_n) \ne f(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_n)$. Then $C(f) \ge n-1$.*

2. *Let $f = (f^{(1)}, \ldots, f^{(m)}) : \mathbb{B}^n \to \mathbb{B}^m$, where $f^{(k)}$ is the k^{th} component of f. If the m component functions $f^{(i)}$ are pairwise different and each of them satisfies $C(f^{(i)}) \geq c \geq 1$ then $C(f) \geq c + m - 1$.*

The proof is given in [12]. Note that for linear functions, statement 1 of Proposition 1 follows from Theorem 1 for $P_n(A) =$ "A has a row with $n+1$ ones".

Idea 2. Suppose that for n steps, there exists an input in the circuit with two outgoing edges, and, moreover, in m of these cases both of these edges go to a gate (rather than a gate and an output). Then $C(f) \geq n + m$.

Theorem 2. *We call a nonzero entry* unique *if it is the only nonzero entry in its row. Suppose that $\mathcal{P} = \{P_n\}_{n=1}^{\infty}$ is a series of predicates defined on matrices over $\mathbb{F}_2$ with the following properties:*

- *if $P_1(A)$ holds then $C(A) \geq 1$;*
- *if $P_n(A)$ holds then $P_m(A)$ holds for every $1 \leq m \leq n$;*
- *if $P_n(A)$ holds then, for every index i, if the i^{th} column has no unique entries then $P_{n-2}(A_{-i})$ holds, otherwise $P_{n-1}(A_{-i})$ holds.*

Then, for every matrix A with $\geq n+1$ different columns, if $P_n(A)$ holds for some n then $C(A) \geq n$ and, moreover, $C_{\frac{3}{4}}(A) \geq n$.

Proof. We argue by induction on n; for $n = 1$ the statement is obvious.

Consider the first gate g in the optimal circuit implementing A. Since g is first, its incoming edges come from the inputs of the circuit, denote them by x_i and x_j. There are three cases.

1. One of the input variables of g, say x_i, goes directly to an output y_k. Then by setting x_i to a constant we can eliminate one gate. however, in this case y_k corresponds to a row with only one nonzero element, so ith colum has a unique element, so $P_{n-1}(A_{-i})$ hold. Therefore, we invoke the induction hypothesis as $C(A_{-i}) \geq n - 1$ and get the necessary bound.

2. One of the input gate of g, say x_i, goes to another gate. Then by setting x_i to a constant we can eliminate two gates, by properties of P_n $P_{n-2}(A_{-i})$ holds, so we invoke the induction hypothesis as $C(A_{-i}) \geq n - 2$.

3. Neither x_i nor x_j enters any other gate or output. In this case, A is a function of neither x_i nor x_j but only $g(x_i, x_j)$; we show that this cannot be the case for a function computing A on more than $\frac{3}{4}$ of the inputs. A itself depends on x_i and x_j separately because all of its columns are different; in particular, for one of these variables, say x_i, there exists an output y_k that depends only on x_i: $y_k = x_i \oplus \bigoplus_{x \in X} x$, where $x_j \notin X$. On the other hand, since every gate in an optimal circuit nontrivially depends on both inputs, there exist values a and b such that $g(0, a) = g(1, b)$. Thus, for every assignment of the remaining variables, either on input strings with $(x_i = 0, x_j = a)$ or on input strings with $(x_i = 1, x_j = b)$ the circuit makes a mistake, which makes it wrong on at least $\frac{1}{4}$ of all inputs. □

Note that Theorem 2 directly generalizes and strengthens Theorem 1.

Corollary 1. *Suppose that $\mathcal{R} = \{R_n\}_{n=1}^{\infty}$ and $\mathcal{Q} = \{Q_m\}_{m=1}^{\infty}$ are two series of predicates defined on matrices over $\mathbb{F}_2$ with the following properties:*

- *if $R_1(A)$ holds then $C(A) \geq 1$;*
- *if $R_n(A)$ holds then $R_k(A)$ holds for every $1 \leq k \leq n$;*
- *if $R_n(A)$ holds then, for every i, $R_{n-1}(A_{-i})$ holds;*
- *if $Q_1(A)$ holds then $C(A) \geq 1$;*
- *if $Q_m(A)$ holds then $Q_k(A)$ holds for every $1 \leq k \leq n$;*
- *if $Q_m(A)$ holds then, for every i, $Q_{m-1}(A_{-i})$ holds;*
- *if $Q_m(A)$ holds and A_{-i} has more zero rows than A (i.e., removing the i^{th} column has removed the last nonzero element from at least one row) then $Q_m(A_{-i})$ holds.*

Then, for every matrix A with $\geq n+1$ columns, all of whose columns are different, if $R_n(A)$ and $Q_m(A)$ hold for some $n \geq m$ then $C(A) \geq n+m$ and, moreover, $C_{\frac{3}{4}}(A) \geq n+m$.

Proof. Immediately follows from Theorem 2 for $P_n(A) = \exists k R_k(A) \wedge Q_{n-k}(A)$. □

Corollary 2 ([12, Lemma 5]). *Let $t, u \geq 1$. Assume that χ is a linear function with matrix A over $\mathbb{F}_2$. Assume also that all columns of A are different, every row of A has at least u nonzero entries, and after removing any t columns of A, the matrix still has at least one row containing at least two nonzero entries. Then $C(\chi) \geq u+t$ and, moreover, $C_{3/4}(\chi) \geq u+t$.*

Proof. Take $P_n(A) =$"After removing any n columns of A, it still has at least one nonzero row", $Q_0(A) =$"true", and $Q_m(A) =$"Every row of A has at least $m+1$ ones" for $m > 0$. Then $P_{t+1}(A)$ and $Q_{u-1}(A)$ hold, and $\mathcal{P}$ and $\mathcal{Q}$ satisfy the conditions of Corollary 1, which gives the desired bound. Note that in this case, Q_m for $m > 0$ cannot hold for a matrix where a row has only a single one, so in the gate elimination proof, for the first $u-1$ steps two gates will be eliminated, and then for $t-u+2$ steps, one gate will be eliminated. □

We also derive another, even stronger corollary that will be important for new feebly secure constructions.

Corollary 3. *Let $t \geq u \geq 2$. Assume that A is a $u \times t$ matrix with different columns, and each column of A has at least two nonzero elements (ones). Then $C(A) \geq 2t-u$ and, moreover, $C_{\frac{3}{4}}(A) \geq 2t-u$.*

Proof. Take $P_n(A) =$"twice the number of nonzero columns in A less the number of nonzero rows in A is at least n". Then $P_{2t-u}(A)$ holds, and $\mathcal{P}$ satisfy the conditions of Theorem 2. □

Naturally, we could prove Corollary 3 directly. We have chosen the path of generalization for two reasons: one, to make Theorem 3 more precise and more general, and two, to show the limits of gate elimination for linear functions. As

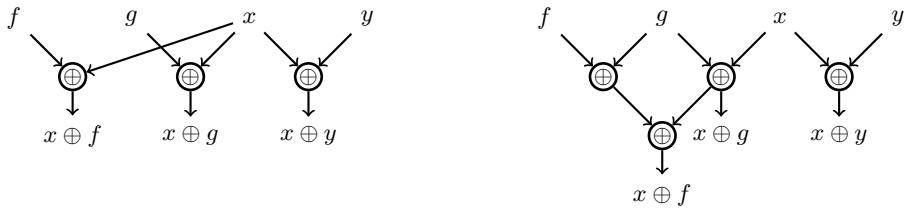

Fig. 1. Rewiring linear circuits.

we have already mentioned, for linear functions we cannot count on nonlinear gates that could eliminate their descendants. In Theorems 1 and 2, we have considered two basic cases: when there is only one edge outgoing from a variable and when there are two edges (going either to two gates or to a gate and an output).

Figure 1 shows why we cannot expect anything more, e.g., a variable with three outgoing edges. On the left, Figure 1 shows a part of a circuit where a variable x has three outgoing edges. On the right, a rewiring of the same circuit that has all the same outputs but x only enters two gates. Naturally, this does not prove anything: we have introduced a new gate (so the circuit is no longer optimal) and have only relocated the extra input from x to an extra input from f. However, this simple example does show that to get better bounds, simple local gate elimination does not suffice, and one has to consider global properties of the function and the corresponding optimal circuit.

We finish this section with an extension of these results to block diagonal matrices. In general, we cannot prove that the direct sum of several functions has circuit complexity equal to the sum of the circuit complexities of these functions; counterexamples are known as "mass production" [26]. However, for linear functions and gate elimination in the flavours of Theorems 1 and 2, we can. The following theorem generalizes Lemma 6 of [12].

Theorem 3. *Suppose that a linear function χ is given by a block diagonal matrix*

$$\begin{pmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & A_k \end{pmatrix},$$

and every A_j satisfies the conditions of Theorem 2 with predicates $\mathcal{P}^j = \{P^j_n\}_{n=1}^{\infty}$, and $P^j_{n_j}(A_j)$ hold for every j. Then $C(\chi) \geq \sum_{j=1}^{k} n_j$.

Proof. We invoke Theorem 2 with the predicate composed of original predicates:

$$P_n = \bigvee_{i_1+\ldots+i_k=n} P^1_{i_1} \wedge P^2_{i_2} \wedge \ldots \wedge P^k_{i_k}.$$

It is now straightforward to check that $\mathcal{P} = \{P_n\}_{n=1}^{\infty}$ satisfies the conditions of Theorem 2 (since every deleted column affects only one block), and the block diagonal matrix satisfies $P_{n_1+\ldots+n_k}$. □

4 A New Linear Feebly Secure Trapdoor Function

In our constructions, we follow the general idea of [12]: first, we find a feebly trapdoor candidate that has the adversary work harder than function inversion but function evaluation is even harder. Then, we add a feebly secure one-way function as a separate block and thus reduce the work needed for function evaluation; this construction has been discussed in detail in [12].

We begin with some preliminaries. By U_n, we denote the upper triangular square $n \times n$ matrix with a bidiagonal inverse:

$$U_n = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}, \qquad U_n^{-1} = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix};$$

note that U_n^2 is an upper triangular matrix with zeros and ones chequered. In what follows, we often write matrices that consist of other matrices as blocks; e.g., $(\, U_n \; U_n \,)$ is an $n \times 2n$ matrix consisting of two upper triangular blocks.

Lemma 1. *1.* $C_{\frac{3}{4}}(U_n) = n - 1$.
2. $C_{\frac{3}{4}}(U_n^2) = n - 2$.
3. $C_{\frac{3}{4}}(U_n^{-1}) = n - 1$.
4. $C_{\frac{3}{4}}((\, U_n \; U_n \,)) = 2n - 1$.
5. $3n - 6 \le C_{\frac{3}{4}}((\, U_n^2 \; U_n \,)) \le C((\, U_n^2 \; U_n \,)) \le 3n - 3$.
6. $3n - 4 \le C_{\frac{3}{4}}((\, U_n \; U_n^{-1} \,)) \le C((\, U_n \; U_n^{-1} \,)) \le 3n - 2$.

Proof. Lower bounds in items 1–3 are trivial: every row is different and no inputs except one (two for 2) are connected to outputs directly. Thus, we need at least one gate per row. The lower bound for item 4 follows from simple counting: the first row of this matrix has $2n$ nonzero entries, so at least $2n - 1$ gates are needed to compute it. The lower bound for item 5 (respectively, 6) follows by Corollary 3: the matrix $(\, U_n^2 \; U_n \,)$ (resp., $(\, U_n \; U_n^{-1} \,)$) satisfies its assumptions except for three (resp., two) columns, so the corollary is invoked for $t = 2n - 3$ (resp., $t = 2n - 2$) and $u = n$.

We prove upper bounds by providing explicit circuit constructions. To compute 1, note that every row differs from the previous one only in a single position, so we can compute each output out_i as $out_{i+1} \oplus in_i$. Moreover, $out_n = in_n$ so we need no gates for it. The same idea applies in 2, but in this case out_n and out_{n-1} are computed directly, and $out_i = out_{i-2} \oplus in_i$. To compute 3, we simply compute each row independently. To compute 4, we apply an idea from [12]. Note that $(\, U_n \; U_n \,) \cdot \binom{a}{b} = U_n \cdot a \oplus U_n \cdot b = U_n \cdot (a \oplus b)$. We use n gates to compute $a \oplus b$ and then compute the result using $n - 1$ gates. To compute 5 and 6, note that $(\, A \; B \,) \cdot \binom{a}{b} = A \cdot a \oplus B \cdot b$. Thus, we can divide each of these computations

in two parts which can be computed independently using previous algorithms, and then use n gates to compute the final XOR. □

For the first construction, we assume that lengths of public information pi, trapdoor information ti, message m, and the cipher c are the same and equal n. We let $ti = U_n \cdot pi$, $c = \left(\begin{smallmatrix} U_n^{-1} & U_n \end{smallmatrix}\right) \cdot \left(\begin{smallmatrix} m \\ pi \end{smallmatrix}\right)$. In this case, an adversary would have to compute the matrix $\left(\begin{smallmatrix} U_n & U_n \end{smallmatrix}\right) \cdot \left(\begin{smallmatrix} c \\ ti \end{smallmatrix}\right) = \left(\begin{smallmatrix} U_n & U_n^2 \end{smallmatrix}\right) \cdot \left(\begin{smallmatrix} c \\ pi \end{smallmatrix}\right)$. Now, inversion without the trapdoor is harder than inversion with trapdoor, but encryption is about the same complexity as inversion without trapdoor, so we cannot call it a feebly trapdoor function yet.

To solve this problem, we consider a feebly one-way linear function A and construct the protocol in the following way (I_n is the identity matrix here):

$$\begin{aligned}
\mathrm{Key}_n &= \left(\begin{smallmatrix} U_n & 0 \\ 0 & I_n \end{smallmatrix}\right) \cdot \left(\begin{smallmatrix} s & s \end{smallmatrix}\right) = \left(\begin{smallmatrix} t_i \\ p_i \end{smallmatrix}\right), \\
\mathrm{Eval}_n &= \begin{pmatrix} U_n^{-1} & U_n & 0 \\ 0 & 0 & A \end{pmatrix} \cdot \begin{pmatrix} m_1 \\ pi \\ m_2 \end{pmatrix} = \left(\begin{smallmatrix} c_1 \\ c_2 \end{smallmatrix}\right), \\
\mathrm{Inv}_n &= \begin{pmatrix} U_n & U_n & 0 \\ 0 & 0 & A^{-1} \end{pmatrix} \cdot \begin{pmatrix} c_1 \\ ti \\ c_2 \end{pmatrix} = \left(\begin{smallmatrix} m_1 \\ m_2 \end{smallmatrix}\right).
\end{aligned}$$

The adversary's problem now becomes to compute

$$\mathrm{Adv}_n = \begin{pmatrix} U_n & U_n^2 & 0 \\ 0 & 0 & A^{-1} \end{pmatrix} \cdot \begin{pmatrix} c_1 \\ pi \\ c_2 \end{pmatrix} = \left(\begin{smallmatrix} m_1 \\ m_2 \end{smallmatrix}\right).$$

For the feebly one-way function A, we fix a small $\epsilon > 0$ and take the Hiltgen's linear function with order of security $2 - \epsilon$ [10]; we take its size to be λn with λ chosen below. In Hiltgen's constructions, it means that $C_{\frac{3}{4}}(A) = \lambda n + o(n)$, and $C_{\frac{3}{4}}(A^{-1}) = (2-\epsilon)\lambda n + o(n)$. Now Lemma 1 and Theorem 3 imply the following complexity bounds:

$$\begin{aligned}
C_{\frac{3}{4}}(\mathrm{Key}_n) &= n - 1, \\
C_{\frac{3}{4}}(\mathrm{Eval}_n) &= 3n + \lambda n + o(n) &&= (3+\lambda)n + o(n), \\
C_{\frac{3}{4}}(\mathrm{Inv}_n) &= 2n + (2-\epsilon)\lambda n + o(n) &&= (2 + (2-\epsilon)\lambda)n + o(n), \\
C_{\frac{3}{4}}(\mathrm{Adv}_n) &= 3n + (2-\epsilon)\lambda n + o(n) &&= (3 + (2-\epsilon)\lambda)n + o(n).
\end{aligned}$$

The order of security of this construction is now

$$\begin{aligned}
\lim_{n\to\infty} \left(\min \left(\frac{C_{3/4}(\mathrm{Adv}_n)}{C(\mathrm{Eval}_n)}, \frac{C_{3/4}(\mathrm{Adv}_n)}{C(\mathrm{Inv}_n)}, \frac{C_{3/4}(\mathrm{Adv}_n)}{C(\mathrm{Key}_n)} \right) \right) = \\
= \min \left(\frac{3 + (2-\epsilon)\lambda}{3+\lambda}, \frac{3 + (2-\epsilon)\lambda}{2 + (2-\epsilon)\lambda} \right).
\end{aligned}$$

This expression reaches maximum for $\lambda = \frac{1}{1-\epsilon}$, and this maximum is $\frac{5-4\epsilon}{4-\epsilon}$, which tends to $\frac{5}{4}$ as $\epsilon \to 0$. Thus, we have proven the following theorem.

Theorem 4. *For every $\epsilon > 0$, there exists a linear feebly trapdoor function with seed length $\mathrm{pi}(n) = \mathrm{ti}(n) = n$, length of inputs and outputs $c(n) = m(n) = 2n$, and order of security $\frac{5}{4} - \epsilon$.*

In Theorem 3, we have generalized a hardness amplification procedure similar to [12, Theorem 2]; with it, we can obtain superpolynomial security guarantees against weaker adversaries.

Theorem 5. *For every $\epsilon > 0$, there exists a linear feebly trapdoor function with seed length $\mathrm{pi}(n) = \mathrm{ti}(n) = n$, length of inputs and outputs $c(n) = m(n) = 2n$, complexities $C_{\frac{3}{4}}(\mathrm{Key}_n) = n-1$, $C_{\frac{3}{4}}(\mathrm{Eval}_n) = 4n+o(n)$, and $C_{\frac{3}{4}}(\mathrm{Inv}_n) = \frac{4-\epsilon}{1-\epsilon}n + o(n)$, and order of security $\frac{5-4\epsilon}{4-\epsilon}$. Moreover, no adversary with less than $\frac{5-4\epsilon}{1-\epsilon}n - \frac{5}{2}\delta\sqrt{n}$ gates can invert this feebly trapdoor function on more than $2^{-\delta\sqrt{n}+o(\sqrt{n})}$ of its inputs for any constant $\delta > 0$.*

Proof. We consider the block diagonal matrix

$$H = \begin{pmatrix} X & 0 & \dots & 0 \\ 0 & X & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & X \end{pmatrix},$$

with m diagonal blocks, where X is the matrix of the trapdoor function constructed in Theorem 4, and apply Theorem 3. Stacking the matrices up in a large block diagonal matrix does not change the parameters of a feebly trapdoor function. □

5 Nonconstructive bounds for linear functions

In Section 3, we have seen that we cannot currently hope to prove more than linear lower bounds on general circuit complexity; the same is true, of course, for linear functions. A classical result shows by counting that among general Boolean functions of n variables, almost all of them have circuit complexity $\geq \frac{1}{n}2^n$. But maybe for the linear case, nonlinear bounds are impossible from the beginning?

It turns out that linear functions with nonlinear bounds do exist. References to this result can be found [2, 4], but we have not been able to find a detailed proof in literature, so we include it here and refine it to get exact constants.

Theorem 6. *1. For every n there exists a constant δ_n such that the circuit complexity of all linear functions $\phi : \{0,1\}^n \to \{0,1\}^n$ does not exceed $\delta_n \frac{n^2}{\log n}$, and $\lim_{n\to\infty} \delta_n = 1$.*

2. For every $n \geq 3$, there exists a linear Boolean function $\phi : \{0,1\}^n \to \{0,1\}^n$ with circuit complexity greater than $\frac{n^2}{2\log n}$.

Proof. 1. *Upper bound.* Let A be the matrix of ϕ. For clarity, we assume n is a power of 2 (the same proof goes through with very minor modifications if n is not). We implement A as follows. First, we generate c_i as all possible rows of zeros and ones of length $l = q \log n$, where q is a constant to be selected later. Denoting the inputs of ϕ by $x = (\, x_1 \, \cdots \, x_n \,)$, we preprocess the values of all

possible combinations of $c_i \cdot \begin{pmatrix} x_{j+1} \\ \cdots \\ x_{j+l} \end{pmatrix}$, where j is a multiple of l. The total number of gates needed for this operation is bounded from above by

$$2^{q \cdot \log n} \cdot \frac{n}{q \cdot \log n} \cdot (q \log n - 1) \leq n^q \cdot n = n^{q+1}.$$

Let $A_1, A_2, \cdots, A_n$ be the columns of A. To find $A \cdot x$, we compute

$$A \cdot x = \begin{pmatrix} A_1 & \ldots & A_l \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_l \end{pmatrix} \oplus \begin{pmatrix} A_{l+1} & \cdots & A_{2l} \end{pmatrix} \begin{pmatrix} x_{l+1} \\ \vdots \\ x_{2l} \end{pmatrix} \oplus \ldots \oplus \begin{pmatrix} A_{n-l+1} & \cdots & A_n \end{pmatrix} \begin{pmatrix} x_{n-l+1} \\ \vdots \\ x_n \end{pmatrix}.$$

After preprocessing, every product in this formula will already be computed, and all we need to do is to choose a correct wire, so no gates are needed here. After that, $n \cdot (\frac{n}{l} - 1)$ gates are needed to XOR for the total result. Thus, the total number of gates needed is

$$n^{q+1} + n \cdot (\frac{n}{q \cdot \log n} - 1) = n^{q+1} + \frac{n^2}{q \cdot \log n} - n.$$

Setting $q = \frac{1}{1+\epsilon}$, we get the upper bound $(1+\epsilon)\frac{n^2}{\log n}$ for an arbitrarily small ϵ.

2. *Lower bound.* The lower bound is proven by counting. Let $q = (1-\epsilon)\frac{n^2}{2 \log n}$. We estimate T, the number of circuits of size $\leq q$. There are 16 types of gates, and every circuit's description consists of type and inputs for every gate. Therefore,

$$T \leq \frac{q \cdot (16 \cdot (n+q)^2))^q}{q!} \leq q \cdot (16e)^q \cdot \frac{(n+q)^{2q}}{q^q} \leq q \cdot (64e)^q \cdot \frac{q^{2q}}{q^q} = (64 \cdot e \cdot q)^q \leq$$

$$\leq (64 \cdot e \cdot q)^{q+1} \leq (n^2)^{(1-\epsilon) \cdot \frac{n^2}{2 \log n} + 1} = (2^{2 \log n})^{(1-\epsilon) \cdot \frac{n^2}{2 \log n} + 1} = 2^{(1-\epsilon)n^2 + 2 \log n}.$$

But the total number of linear Boolean functions of n arguments is 2^{n^2}, which is greater than $2^{(1-\epsilon)n^2 + 2 \log n}$. Therefore, there are Boolean functions with circuit complexity exceeding $\frac{n^2}{2 \log n}$. □

6 Conclusion

In this paper, we have discussed in detail the circuit complexity of linear Boolean functions. We have proven two general statements on gate elimination in linear functions, derived several corollaries, and applied them to find a new linear feebly secure trapdoor function, with better order of security than the known one [12]. While feebly secure cryptographic primitives can hardly be put to any practical use, they are still important from the theoretical point of view. As sad as it sounds, this is actually the frontier of provable, mathematically sound results on security; we do not know how to prove anything stronger. However, in Section 5 we have seen that with these bounds, we are only scratching the surface even for linear Boolean functions, let alone nonlinear ones.

Further work in this direction is twofold. One can further develop the notions of feebly secure primitives. Orders of security can certainly be improved; perhaps, new primitives (key agreement protocols, zero knowledge proofs etc.) can find their feebly secure counterparts. This work can widen the scope of feebly secure methods, but the real breakthrough can only come from one place.

It becomes clear that cryptographic needs call for further advances in general circuit complexity. General circuit complexity has not had a breakthrough since the 1980s; nonconstructive lower bounds are easy to prove by counting, but constructive lower bounds remain elusive. The best bound we know is $3n - o(n)$, proven in 1984 [3]. At present, we do not know how to rise to this challenge; none of the known methods seem to work, so a general breakthrough is required for nonlinear lower bounds on circuit complexity. The importance of such a breakthrough can hardly be overstated; feebly secure cryptographic constructions provide yet another application for new circuit lower bounds.

Acknowledgements

We thank Olga Melanich and the anonymous referees for pointing out important shortcomings in a preliminary version of our paper.

Work of the first author has been supported by the Yandex Stipend Program. Work of the second author has been supported by the Russian Presidential Grant Programme for Young Ph.D.'s, grant no. MK-4089.2010.1, for Leading Scientific Schools, grant no. NSh-5282.2010.1, Federal Target Programme "Scientific and scientific-pedagogical personnel of the innovative Russia", and Russian Fund for Basic Research grants.

References

1. Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: Proceedings of the 29$^{\text{th}}$ Annual ACM Symposium on Theory of Computing. pp. 284–293 (1997)
2. Alon, N., Karchmer, M., Wigderson, A.: Linear circuits over GF(2). SIAM Journal of Computing 19(6), 1064–1067 (1990)
3. Blum, N.: A boolean function requiring $3n$ network size. Theoretical Computer Science 28, 337–345 (1984)
4. Bublitz, S.: Decomposition of graphs and monotone formula size of homogeneous functions. Acta Informatica 23, 410–417 (1986)
5. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Transactions on Information Theory IT-22, 644–654 (1976)
6. Dwork, C.: Positive applications of lattices to cryptography. In: Proceedings of the 22$^{\text{nd}}$ International Symposium on Mathematical Foundations of Computer Science, Lecture Notes in Computer Science. vol. 1295, pp. 44–51 (1997)
7. Goldreich, O.: Foundations of Cryptography. Basic Tools. Cambridge University Press (2001)
8. Grigoriev, D., Hirsch, E.A., Pervyshev, K.: A complete public-key cryptosystem. Groups, Complexity, and Cryptology 1, 1–12 (2009)

9. Harnik, D., Kilian, J., Naor, M., Reingold, O., Rosen, A.: On robust combiners for oblivious transfers and other primitives. In: Proceedings of EuroCrypt 05, Lecture Notes in Computer Science. vol. 3494, pp. 96–113 (2005)
10. Hiltgen, A.P.: Constructions of feebly-one-way families of permutations. In: Proc. of AsiaCrypt '92. pp. 422–434 (1992)
11. Hiltgen, A.P.: Cryptographically Relevant Contributions to Combinational Complexity Theory, ETH Series in Information Processing, vol. 3. Konstanz: Hartung-Gorre (1994)
12. Hirsch, E.A., Nikolenko, S.I.: A feebly secure trapdoor function. In: Proceedings of the 4^{th} Computer Science Symposium in Russia, Lecture Notes in Computer Science. vol. 5675, pp. 129–142 (2009)
13. Koblitz, N., Menezes, A.: Another look at "provable security". Journal of Cryptology 20(1), 3–37 (2007)
14. Kojevnikov, A.A., Nikolenko, S.I.: New combinatorial complete one-way functions. In: Proceedings of the 25^{th} Symposium on Theoretical Aspects of Computer Science. pp. 457–466. Bordeaux, France (2008)
15. Kojevnikov, A.A., Nikolenko, S.I.: On complete one-way functions. Problems of Information Transmission 45(2), 108–189 (2009)
16. Lamagna, E.A., Savage, J.E.: On the logical complexity of symmetric switching functions in monotone and complete bases. Tech. rep., Brown University, Rhode Island (July 1973)
17. Melanich, O.: Nonlinear feebly secure cryptographic primitives. PDMI preprints 12 (2009)
18. Paul, W.J.: A $2,5n$ lower bound on the combinational complexity of boolean functions. SIAM Journal of Computing 6, 427–443 (1977)
19. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the 37^{th} Annual ACM Symposium on Theory of Computing. pp. 84–93 (2005)
20. Regev, O.: Lattice-based cryptography. In: Proceedings of the 26^{th} Annual International Cryptology Conference (CRYPTO'06), Lecture Notes in Computer Science. vol. 4117, pp. 131–141 (2006)
21. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
22. Savage, J.E.: The Complexity of Computing. Wiley, New York (1976)
23. Shannon, C.E.: Communication theory of secrecy systems. Bell System Technical Journal 28(4), 656–717 (1949)
24. Stockmeyer, L.: On the combinational complexity of certain symmetric boolean functions. Mathematical Systems Theory 10, 323–326 (1977)
25. Vernam, G.S.: Cipher printing telegraph systems for secret wire and radio telegraphic communications. Journal of the IEEE 55, 109–115 (1926)
26. Wegener, I.: The Complexity of Boolean Functions. B. G. Teubner, and John Wiley & Sons (1987)