The Sum of PRPs is a Secure PRF

Stefan Lucks[⋆]

Theoretische Informatik, Universität Mannheim
68131 Mannheim, Germany
lucks@th.informatik.uni-mannheim.de

Abstract. Given d independent pseudorandom permutations (PRPs) $\pi_i, \ldots, \pi_d$ over $\{0,1\}^n$, it appears natural to define a pseudorandom function (PRF) by adding (or XORing) the permutation results: $\textsc{sum}^d(x) = \pi_1(x) \oplus \cdots \oplus \pi_d(x)$. This paper investigates the security of $\textsc{sum}^d$ and also considers a variant that only uses one single PRP over $\{0,1\}^n$.

1 Introduction

Cryptography requires an encryption function to be invertible: Someone knowing the (secret) key must be able to recover the plaintext from the ciphertext. Accordingly, under a fixed key, a n-bit block cipher is a permutation $\pi : \{0,1\}^n \rightarrow \{0,1\}^n$. The classical security requirement is that π must behave pseudorandomly, i.e. must be un-distinguishable from a random permutation over $\{0,1\}^n$ without knowing the secret key.

In practice, block ciphers are used in many different modes of operations, and not all of them need an invertible cipher. Sometimes, being invertible can even *hinder* the security of schemes using the cipher. One such example is the "cipher block chaining" (CBC) mode, a standard mode of operation for block ciphers: if more than about $2^{n/2}$ blocks are encrypted, the ciphertext leaks information about the plaintext [2]. So why not simply use a dedicated pseudorandom function (PRF) instead of a pseudorandom permutation (PRP) in such cases? Two reasons are:

- Applications may need both invertible ciphers *and* schemes where the cipher better would not be invertible. Double-using one primitive to implement both is less expensive in terms of memory or chip space.
- There exist quite a lot of "proven" block ciphers, i.e., block ciphers published years ago, intensively cryptanalysed and widely trusted today. There are not as many good candidates for dedicated PRFs.

Hence, instead of constructing pseudorandom functions from scratch, we consider creating them using pseudorandom permutations as underlying building

[⋆] Supported by DFG grant Kr 1521/3-1.

blocks. Recently, the question of how to do this has caught the attention of the cryptographic community [5, 7]. Let $\pi_1, \ldots, \pi_d$ denote random permutations over $\{0,1\}^n$ and $\oplus$ the bit-wise XOR. Bellare, Krovetz and Rogaway [5] point out that the construction $\textsc{sum}^2(x) = \pi_1(x) \oplus \pi_2(x)$ has not (yet) been analysed. In the current paper, we generalise this and analyse $\textsc{sum}^d : \{0,1\}^n \to \{0,1\}^n$ with $\textsc{sum}^d(x) = \pi_1(x) \oplus \cdots \oplus \pi_d(x)$.

Organisation of this Paper:
Section 2 and Section 3 present the notation and the basic definitions we use in this paper and describe some previous research. Section 4 describes the security of the PRF $\textsc{sum}^d(x) = \bigoplus_{1 \le i \le d} \pi_d(x)$. In the following section, we analyse the variant $\textsc{twin}^d : \{0,1\}^{n - \lceil \log_2(d) \rceil} \to \{0,1\}^n$ with $\textsc{twin}^d(x) = \pi(dx) \oplus \cdots \oplus \pi(dx + d - 1)$. Section 6 provides some comments and conclusions. For better tangibility, the appendix considers the two-dimensional special case $\textsc{sum}^2$.

2 Preliminaries

We write $\mathbb{F}_{m,n}$ for the set of all functions $\{0,1\}^m \to \{0,1\}^n$ and $\mathbb{F}_n = \mathbb{F}_{n,n}$. For choosing a random value x, uniformly distributed in a set M, we write $x \in_{\mathsf{R}} M$. A random function $\psi \in \mathbb{F}_{m,n}$ is a function $\psi \in_{\mathsf{R}} \mathbb{F}_{m,n}$. If S_n is the set of all permutations in $\mathbb{F}_n$, a random permutation over $\{0,1\}^n$ is a function $\pi \in_{\mathsf{R}} S_n$.

To measure the "pseudorandomness" of a function $f \in \mathbb{F}_{m,n}$, chosen "somehow randomly" but in general not uniformly distributed, we consider an adversary A trying to distinguish between f and a random function $R \in_{\mathsf{R}} \mathbb{F}_{m,n}$. A has access to an oracle Q. A chooses inputs $x \in \{0,1\}^n$; Q responds $Q(x) \in \{0,1\}^n$. Q either simulates $R \in_{\mathsf{R}} \mathbb{F}_{m,n}$, or f. A's output is $A(Q) \in \{0,1\}$. We view A as a probabilistic algorithm, hence the output $A(Q)$ is a random variable over $\{0,1\}$. $A(Q)$ depends on the random choice of f and the internal coin flips of A and Q. We evaluate the (unsigned) difference of the probabilities $\mathrm{pr}[A(Q) = 1]$ for $Q = R$ and $Q = f$, i.e. A's "PRF advantage" $\mathrm{Adv}^{\mathrm{Fun}}_{A,f}$ with respect to f:

$$\mathrm{Adv}^{\mathrm{Fun}}_{A,f} = |\mathrm{pr}[A(R) = 1] - \mathrm{pr}[A(f) = 1]|.$$

A's "PRP advantage" $\mathrm{Adv}^{\mathrm{Perm}}_{A,\pi}$ is defined similarly. Here, the oracle Q simulates a random permutation $P \in_{\mathsf{R}} S_n$ and $\pi \in S_n$.

$$\mathrm{Adv}^{\mathrm{Perm}}_{A,\pi} = |\mathrm{pr}[A(P) = 1] - \mathrm{pr}[A(\pi) = 1]|.$$

Definition 1. *A function $f \in \mathbb{F}_{m,n}$ is a (q,a)-secure PRF, if all adversaries A asking at most q oracle queries are restricted to $\mathrm{Adv}^{\mathrm{Fun}}_{A,f} \le a$. Similarly, we define a (q,a)-secure PRP π: $\mathrm{Adv}^{\mathrm{Perm}}_{A,\pi} \le a$.*

Note that "ideal" schemes are $(\infty, 0)$-secure: a random function is a $(\infty, 0)$-secure PRF, and a random permutation is a $(\infty, 0)$-secure PRP.

The notion of "(q,a)-security" is very strong, since the adversaries' running time is not limited. By simply searching the key-space, one could easily distinguish a block cipher from a random permutation. We claim that one can *approximatively* describe a practically secure block cipher under a random key as an $(\infty, 0)$-secure PRP, see Section 6.1.

We interchangeably view b-bit strings $s = (s_{b-1}, \dots, s_0) \in \{0,1\}^b$ as b-bit numbers $s = \sum_{0 \le i < b} s_i * 2^i$.

3 Previous Work

3.1 Using a PRP as PRF

It is widely known that a random permutation over $\{0,1\}^n$ is a $(q, q^2/2^n)$-secure PRF. Since it nicely fits to our later results, we formalise this here:

Theorem 1. *The random permutation $\pi \in \mathbb{F}_n$ is a (q, a)-secure PRF with $a = q^2/2^{n+1}$. An adversary A^* exists to distinguish π from a random function with an advantage of $\mathrm{Adv}^{\mathrm{Fun}}_{A^*,\pi} = \theta(q^2/2^n)$.*

Proof: [Sketch] If by chance a random function R behaves like a permutation, i.e., for all q pairs $(x_1, R(x_1))$ no collision $R(x_i) = R(x_j)$ with $x_i \neq x_j$ occurs, then no adversary can distinguish between R and a random permutation. On the other hand, any collision proves that R is no permutation. With q inputs, the probability to get a collision is $2^{-n} \sum_{1 \le i < q} i \le q^2/2^{n+1}$. □

Theorem 1 justifies to use a block cipher (i.e. a PRP) as a PRF – if the famous *birthday bound* $q \ll 2^{n/2}$ is observed. What about $q > 2^{n/2}$? Note that the function $f^{\oplus}$ with $f^{\oplus}(x) = \pi(x) \oplus x$ is unlikely to be invertible, but is not a better PRF since $\pi(x) = f^{\oplus}(x) \oplus x$ [7, 5].

3.2 Using simple operations and PRFs as Building Blocks

Much research dealt with constructing complex cryptographic operations from (seemingly) simple ones: Levin [8] constructed "pseudorandom bit generators" from "one-way functions", Goldreich, Goldwasser, and Micali [6] constructed PRFs from "pseudorandom bit generators", and Luby and Rackoff [9] constructed PRPs from PRFs. A lot of work has been done on improvements of the Luby-Rackoff construction, some recent examples are [10–12]. Now we are going into the opposite direction: We construct PRFs from PRPs.

Another direction of cryptographic research was how to construct PRFs from smaller PRFs. Aiello and Venkatesan [1] presented a construction for PRFs over $\{0,1\}^{2n}$ using PRFs over $\{0,1\}^n$ as building blocks.

3.3 Constructing a PRF from PRPs

"**Data dependent re-keying**" was proposed by Bellare, Krovetz, and Rogaway [5]. Here, a block cipher E with k-bit keys is a family of 2^k independent random permutations. Set $j := \lceil k/n \rceil$. For keys $K_1, \dots, K_j \in \{0,1\}^k$, the function $f^{\mathrm{BKR}}_{K_1,\dots,K_j}$ maps $x \in \{0,1\}^n$ to $f^{\mathrm{BKR}}_{K_1,\dots,K_j}(x) \in \{0,1\}^n$ by the following algorithm:

$$\begin{array}{lll} K' & := E_{K_1}(x)||\cdots||E_{K_j}(x); & (* \text{ Concatenate the values } E_{k_i}(x). *) \\ K'' & := K' \bmod 2^k; & (* \text{ We only need } k \text{ of } nj \geq k \text{ bits. } *) \\ f^{\mathrm{BKR}}_{K_1,\ldots,K_j}(x) & := E_{K''}(x); & (* \text{ Use the derived key } K'' \text{ to encrypt the input. } *) \end{array}$$

In a formal model, data dependent re-keying is provably more secure than simply using one PRP as a PRF [5]. The model is based on the adversary having access to the block cipher E by asking additional oracle queries: choose keys $K \in \{0,1\}^k$ and texts $T \in \{0,1\}^n$ and ask the oracle for $E_K(T)$ and $E^{-1}(T)$. [5, Theorem 5.2] indicates that $f^{\mathrm{BKR}}_{K_1,\ldots,K_j}$ is a (t,q,a)-secure PRF with $a \approx 0$ if $t \ll \min\{2^{4k/5}, 2^n\}$ and $q \ll \min\{2^{4k/5}, 2^{(n+k)/2}\}$. A variation of this scheme speeds up counter mode encryption: For a small constant d, the same K'' is used for 2^d steps.

Hall et. al. [7] examine two constructions. Let $d \in \{0,\ldots,n\}$ and π be a PRP over $\{0,1\}^n$. The "**truncate**" construction is defined by $f_d^{\mathrm{tr}} : \{0,1\}^n \rightarrow \{0,1\}^{n-d}$ by $f_d^{\mathrm{tr}}(x) = \pi(x) \operatorname{div} 2^d$. The PRF f_d^{tr} is provably secure if $q \ll \min\{2^{(n+d)/2}, 2^{2(n-d)/3}\}$ [7], i.e. if $q \ll 2^{4n/7}$ for $d \approx n/7$.

Given $d \in \{0,\ldots,n\}$ and a PRP π over $\{0,1\}^n$, the **order** construction realizes a PRF $f_d^{\mathrm{ord}} : \{0,1\}^{n-d} \rightarrow S_{2^d}$. Here, S_{2^d} denotes the set of permutations over 2^d elements. The function f_d^{ord} maps $x \in \{0,1\}^{n-d}$ to $f_d^{\mathrm{ord}}(x) \in S_{2^l}$ by sorting the 2^d values $\pi(0\cdots000||x)$, $\pi(0\cdots001||x)$, $\ldots$, $\pi(1\cdots111||x)$.[1] The order construction provably preserves the full security of π: if π is a $(\infty,0)$-secure PRP, then f_d^{ord} is a $(\infty,0)$-secure PRF. On the other hand, the order construction is quite slow, since computing $f_d^{\mathrm{ord}}(x)$ takes 2^d invocations of π.

Recently, Bellare and Impagliazzo [3] described a **general probabilistic lemma** to upper bound the advantage of an adversary in distinguishing between two families of functions.[2]

As an example for applying their general technique, they consider converting a PRP into a PRF. They analyse SUM^2, the two-dimensional special case of the SUM^d-construction we consider in the current paper. They also apply their general technique to analyse two more PRP→PRF constructions: the TWIN^2 variant of SUM^2 (not using the name "TWIN^2"), and the truncate construction from [7].

4 The Construction $\mathbf{SUM}^{d}(x) = \bigoplus_{i=1}^{d} \pi_i(x)$

Consider $d \geq 1$ permutations $\pi_1, \ldots, \pi_d$, we define SUM$^d \in \mathbb{F}_n$ by

$$\textsc{sum}^d(x) = \pi_1(x) \oplus \cdots \oplus \pi_d(x).$$

In the appendix, we regard the the two-dimensional special case SUM^2. The proof of Theorem 5 in the appendix is similar to the proof of Theorem 2 in this

[1] In fact, [7] deals with a function $f_d^{\mathrm{ord}*} : \{0,1\}^{n-d} \rightarrow \{0,1\}^{2^d-1}$. Note that 2^{2^d-1} is the largest power of two dividing $(2^d)! = |S_{2^d}|$ [7, Lemma 1]. Computing $f_d^{\mathrm{ord}*}$ requires 2^d invocations of π and $2^d - 1$ comparisons.

[2] When the current paper was originally written, its author was unaware of [3]. An anonymous referee provided the reference.

section, but requires less technical details. It may be instructive for the reader to first skip to the appendix at page 488 and work through the proof of Theorem 5, and then to continue with the current section.

Theorem 2. *For $d \geq 1$ random permutations $\pi_1, \dots, \pi_d \in \mathbb{F}_n$ and $q \leq 2^{n-1}/d$ is the function* $\textsc{sum}^d$ *a (q, a)-secure PRF with*

$$a \leq 2^{-d(n-1)} * \sum_{0 \leq i < q} i^d.$$

The proof of Theorem 2 requires some technical definitions and lemmas provided below. Set $N := \{0,1\}^n$.

Definition 2. *The set $T \subseteq N^d$ is* "fair"*, if for every $y \in N$*

$$\left| \{ (x_1, \dots, x_d) \in T \mid x_1 \oplus \cdots \oplus x_d = y \} \right| = \frac{|T|}{|N|} = \frac{|T|}{2^n}.$$

If $(x_1, \dots, x_d) \in_{\mathsf{R}} T$, then $y = x_1 \oplus \cdots \oplus x_d$ is a uniformly distributed random value in N if and only if T is fair. To deal with sets that may be unfair, we also define a measurement of being "almost fair".

Definition 3. *$T \subseteq N^d$ is* "z-fair"*:*

- *If a set $V \subseteq N^d$ exists with $|V| = z$ and $V \cap T = \{\}$, such that $V \cup T$ is fair. We call V a "completion set" (short: "c-set") for T.*
- *Or if a set $U \subseteq T$ with $|U| = z$ exists (an "overhanging set" or "o-set"), such that $T - U$ is fair. We also say: T is "z-overhanging-fair".*

Lemma 1.

(a) *Consider the sets $A \subseteq N^a$ and $B \subseteq N^b$. If either A or B or both are fair, then $A \times B \subseteq N^{ab}$ is fair, too.*
(b) *If the two sets $B \subseteq A \subseteq N^d$ are fair, then so is $A - B$.*
(c) *If A is fair and $B \subseteq A$, then $A - B$ is $|B|$-fair.*
(d) *If the two sets $A \subseteq N^d$ and $B \subseteq N^d$ are fair and $|A| \geq |B|$, then $A - B = A \cap \overline{B}$ is $|B - A|$-overhanging-fair.*

Proof: The proofs of (a) and (c) are trivial. Regarding (b), note that A is fair: $|\ \{ (x_1, \dots, x_d) \in A \mid x_1 \oplus \cdots \oplus x_d = y \}\ | = |A|/2^n$ for every $y \in 2^n$. Similarly: $|\ \{ (x_1, \dots, x_d) \in B \mid x_1 \oplus \cdots \oplus x_d = y \}\ | = |B|/2^n$. Thus we get $|\ \{ (x_1, \dots, x_d) \in A \mid (x_1, \dots, x_d) \notin B$ and $x_1 \oplus \cdots \oplus x_d = y \}\ | = |A - B|/2^n$, hence $A - B$ is fair.

To show (d), consider a fair set $B^* \subseteq A$ with $|B^*| = |B|$. B^* contains the elements $x \in (A \cap B)$, and, for every $(x_1, \dots, x_d) \in (\overline{A} \cap B)$, the set B^* contains a unique representative $(y_1, \dots, y_d) \in (A \cap \overline{B})$ with $x_1 \oplus \cdots \oplus x_d = y_1 \oplus \cdots \oplus y_d$. Note that such a set B^* exists since $|B| = |B^*| \leq |A|$ and both A and B are fair. By $R \subseteq B^*$, we denote the set of such representatives, I.e., $|R| = |\overline{A} \cap B|$. Since $A - B^* = (A - B) - R$ is fair, i.e., $A - B$ is $|R|$-overhanging-fair. □

Lemma 2. *Consider the sets $T' \subseteq N^{d-1}$ and $T'' \subseteq N$. Let $z'' = 2^n - |T''|$ (hence T'' is z''-fair). Let $T = T' \times T''$ and $|T| \geq z'z''$. If T' is z'-fair, then T is $z'z''$-fair. More exactly:*

(a) *If $V' \subseteq N^{d-1}$ with $|V'| = z'$ is a c-set for T', then an o-set $U \subseteq T$ for T exists with $|U| = z'z''$.*

(b) *If $U' \subseteq T'$ with $|U'| = z'$ is an o-set for T', then a c-set $V \subseteq N^d$ of size $|V| = z'z''$ exists for T.*

Proof: Note that $V'' = N - T''$ is a c-set for T'' with $|V''| = z''$.

For (a), let $V' \subseteq N^{d-1}$ with $|V'| = z'$ be a c-set for T'. Due to Lemma 1(a), both sets $T' \times (T'' \cup V'')$ and $(T' \cup V') \times V''$ are fair, and

$$\begin{aligned} T' \times T'' &= \quad (T' \times (T'' \cup V'')) \quad - \quad ((T' \cup V') \times V'') \\ &= (T' \times T'') \cup (T' \times V'') - ((T' \times V'') \cup (V' \times V'')) \\ &= \qquad (T' \times T'') \qquad - \qquad (V' \times V''). \end{aligned}$$

Since $|T' \times T''| = |T| \geq z'z'' = |V' \times V''|$ and thus $|T' \times (T'' \cup V'')| \geq |(T' \cup V') \times V''|$, we can apply Lemma 1(d) and conclude: $T' \times T''$ is $|V' \times V''|$-fair, and $|V' \times V''| = z'z''$. Also, an o-set of size $|V' \times V''| = z'z''$ exists for $T' \times T''$.

Regarding (b), consider the o-set $U' \subseteq T'$ with $|U'| = z'$. As above, we argue that the sets $T' \times (T'' \cup V'')$ and $(T' - U') \times V''$ are fair, and

$$\begin{aligned} &(T' \times T'') \cup (U' \times V'') \\ &\qquad = \quad (T' \times (T'' \cup V'')) \quad - \quad ((T' - U') \times V'') \\ &\qquad = (T' \times T'') \cup (T' \times V'') - ((T' \times V'') - (U' \times V'')). \end{aligned}$$

Since $((T' - U') \times V'') \subseteq T' \times V'' \subseteq T' \times (T'' \cup V'')$, we can apply Lemma 1(b): the set $(T' \times T'') \cup (U' \times V'')$ is fair. By Lemma 1(c) we find that $T' \times T''$ is $|U' \times V''|$-fair. Especially, $U' \times V''$ is a c-set for $T = T' \times T''$. □

Proof: [of Theorem 2] Our adversary asks $q \leq 2^{n-1}/d$ oracle queries. We write $x_1, \ldots, x_q$ for the inputs chosen by the adversary and $y_1, \ldots, y_q$ for the oracle's corresponding outputs. W. l. o. g., we assume $x_i \neq x_j$ for $i \neq j$. Evaluating SUM^d on these inputs may be thought of as choosing q values $\pi_k(x_1)$, $\ldots$, $\pi_k(x_q)$ for every $k \in \{1, \ldots, d\}$. Since π_k is a random permutation over $\{0,1\}^n$, the values $\pi_k(x_1), \ldots, \pi_k(x_q)$ are random values in $N = \{0,1\}^n$, except that $\pi_k(x_i) \neq \pi_k(x_j)$ for $i \neq j$. We simply write $\pi_{k,j}$ for $\pi_k(x_j)$. Now, generating the random values $y_i = \text{SUM}^d(x_i)$ may be thought of as choosing $\pi_{k,i} \in_{\text{R}} N - \{\pi_{k,1}, \ldots, \pi_{k,i-1}\}$ for $k \in \{1, \ldots, d\}$ and evaluating $y_i = \pi_{1,i} \oplus \cdots \oplus \pi_{d,i}$. We may as well regard this as choosing the d-tuple $\pi_{1,i}, \ldots, \pi_{d,i} \in_{\text{R}} T_i \subseteq N^d$, where T_i is the set of all d-tuples still avaliable, i.e., $T_1 = N^d$ and $T_{i+1} \subseteq T_i$, or exactly:

$$\begin{aligned} T_{i+1} \; = \; N^d \; &- \quad (\{\pi_{1,1}, \ldots, \pi_{1,i}\} \times N^{d-1}) \\ &- (N \times \{\pi_{2,1}, \ldots, \pi_{2,i}\} \times N^{d-2}) \\ &- \qquad\qquad \cdots \\ &- \quad (N^{d-1} \times \{\pi_{d,1}, \ldots, \pi_{d,i}\}) \end{aligned} \tag{1}$$

Note that $|T_{i+1}| \geq 2^{dn} - (di * 2^{(d-1)n})$. We can simulate the generation of the values y_j as follows:

```
For $i := 1$ to $q$: choose $t_i = (\pi_{1,i}, \ldots, \pi_{d,i}) \in_{\mathsf{R}} T_i$;
                     output $y_i = \pi_{1,i} \oplus \cdots \oplus \pi_{d,i}$.
```

The sets T_{i+1} are i^d-fair:

We set $j := i + 1$ and show that if d is odd, a c-set V_j exists for T_j with $|V_j| = (j-1)^d$, and, if d is even, an o-set U_j for T_j of size $|U_j| = (j-1)^d$ exists.

We prove this by induction. If $d = 1$, $V_j = \{y_1, \ldots, y_{j-1}\}$ is a c-set for T_j and $|V_j| = (j-1)^1$. For $d > 1$, we split the d-tuples $(\pi_{j,1}, \ldots, \pi_{j,d}) \in T_j$ up into a $(d-1)$-tuple $(\pi_{j,1}, \ldots, \pi_{j,d-1}) \in T'_j \subseteq N^{d-1}$ and a single value $\pi_{j,d} \in T''_j = N - V''_j$ with $V''_j = \{\pi_{d,1}, \ldots, \pi_{d,j-1}\}$. We know that T''_j is $(j-1)$-fair. Note that $j \le q \le 2^{n-1}/d$, hence $|T'_j| \ge (j-1)^{d-1}$, $|T''_j| \ge j-1$, and, by induction, $|T_j| \ge (j-1)^d$. This will allow us to apply Lemma 2.

Let d be even. Then $d-1$ is odd. Assume that a c-set V'_j for T'_j exists of size $|V'_j| = (j-1)^{d-1}$. The claim follows from Lemma 2(a).

Now, let d be odd. Assume that T'_j is $(j-1)^{d-1}$-fair, and that an o-set $U'_j \subseteq T'_j$ exists with $|U'_j| = (j-1)^d$. The claim follows from Lemma 2(b).

Choosing the d-tuples $(\pi_{1,i}, \ldots, \pi_{d,i})$ from fair sets:

Since we know that c-sets or o-sets of size $(i-1)^d$ for T_i exist, we can simulate the generation of the y_i as described in Figure 1. Either, d is odd and a c-set V_i for T_i exists, or d is even and an o-set U_i exists.

```
Even $d$:

Set bad := 0;
for $i := 1$ to $q$:
    determine $T_i$ and $U_i$;
    choose $t_i = (\pi_{1,i}, \ldots, \pi_{d,i}) \in_{\mathsf{R}} T_i$;
    if $t_i \in U_i$ then
        bad := 1;
    output $y_i := \pi_{1,i} \oplus \cdots \oplus \pi_{d,i}$;
output bad.
```

```
Odd $d$:

Set bad := 0;
for $i := 1$ to $q$:
    determine $T_i$ and $V_i$; $S_i := T_i \cup V_i$;
    choose $t_i = (\pi_{1,i}, \ldots, \pi_{d,i}) \in_{\mathsf{R}} S_i$;
    if $t_i \notin T_i$ then
        bad := 1;
        choose $t_i = (\pi_{1,i}, \ldots, \pi_{d,i}) \in_{\mathsf{R}} T_i$;
    output $y_i := \pi_{1,i} \oplus \cdots \oplus \pi_{d,i}$;
output bad.
```

Fig. 1. Two simulations for the PRF $\textsc{sum}^d$

When the output $y_i := \pi_{1,i} \oplus \cdots \oplus \pi_{d,i}$ is generated, the d-tuple $t_i = (\pi_{1,i}, \ldots, \pi_{d,i})$ is a uniformly distributed random value $t_i \in_{\mathsf{R}} T_i$. The simulation generates an additional value bad $\in \{0, 1\}$. If bad $= 0$, all t_i are uniformly distributed random d-tuples chosen from fair sets, and thus $y_i \in_{\mathsf{R}} N$. Thus, the advantage of every adversary is at most pr[bad $= 1$].

Evaluating pr[bad $= 1$]:

The simulation in Figure 1 outputs bad $= 1$ if and only if the then-clause is executed at least once, i.e., $\text{pr}[\text{bad} = 1] \leq \sum_{1 \leq i \leq q} \text{pr}[\text{then}]$. We get

$$\text{pr}[\text{then}] = \left\{ \begin{array}{ll} (i-1)^d/|T_i| & \text{for even } d \\ (i-1)^d/(|T_i| + (i-1)^d) & \text{for odd } d \end{array} \right\} \leq \frac{(i-1)^d}{|T_i|}$$

Since $|T_{i+1}| \geq 2^{dn} - (di * 2^{(d-1)n})$ and $i \leq q \leq 2^{n-1}/d$, $|T_{i+1}| \geq 2^{dn} - 2^{dn-1} = 2^{dn-1}$ and thus

$$\frac{(i-1)^d}{|T_i|} \leq \frac{(i-1)^d}{2^{dn-1}}$$

$$\Rightarrow \quad \text{pr}[\text{bad} = 1] \leq \sum_{1 \leq i \leq q} \text{pr}[\text{then}] \leq \sum_{1 \leq i \leq q} \frac{(i-1)^d}{2^{dn-1}} \leq \frac{1}{2^{dn-1}} \sum_{0 \leq i < q} i^d.$$

Hence $a \leq \text{pr}[\text{bad} = 1] \leq 2^{-dn+1} \sum_{0 \leq i < q} i^d$. □

Note that $\sum_{1 \leq i < q} i^d = \theta(i^{d+1})$, hence $\text{Adv}^{\text{Fun}}_{A,\text{SUM}^d} \leq \theta(q^{d+1}/2^{nd})$. Depending on d, we provide some examples. For every adversary A, we get:

$$d = 1\text{:} \quad \sum_{0 \leq i < q} i = \frac{q(q-1)}{2} \leq \frac{q^2}{2} \;\Rightarrow\; \text{Adv}^{\text{Fun}}_{A,\text{SUM}^1} \leq \frac{q^2}{2^n} \tag{2}$$

$$d = 2\text{:} \quad \sum_{0 \leq i < q} i^2 = \frac{2q^3 - 3q^2 + q}{6} \leq \frac{q^3}{3} \;\Rightarrow\; \text{Adv}^{\text{Fun}}_{A,\text{SUM}^2} \leq \frac{q^3}{3 * 2^{2n-1}} \tag{3}$$

$$d = 3\text{:} \quad \sum_{0 \leq i < q} i^3 = \frac{q^2(q-1)^2}{4} \leq \frac{q^4}{4} \;\Rightarrow\; \text{Adv}^{\text{Fun}}_{A,\text{SUM}^3} \leq \frac{q^4}{2^{3n+1}}. \tag{4}$$

In general, SUM^d is secure against adversaries asking $q \ll \sqrt[d+1]{2^{dn-1}}$ queries. If a pessimistic estimate of q gives a value $q \ll 2^n$, we can choose d accordingly. In practice, d will be small, e.g., $d \leq 10$.

5 The Construction $\text{TWIN}^d(x) = \bigoplus_{i=0}^{d-1} \pi(dx+i)$

The SUM^d-construction requires d independent PRPs $\pi_1, \ldots, \pi_d$. We may use one block cipher running under d different keys to implement the π_i. Depending on our choice of block cipher and on hardware limitations, frequently changing between encryption under d different keys may be costly, though. Can we construct a secure PRF using a single PRP π over $\{0,1\}$? Consider the function $\text{TWIN}^d : \{0,1\}^{n - \lceil \log_2(d) \rceil} \rightarrow \{0,1\}^n$:

$$\text{TWIN}^d(x) = \pi(dx) \oplus \cdots \oplus \pi(dx + d - 1).$$

(Recall that we interchangeably view b-bit strings $s \in \{0,1\}^b$ as numbers $s \in \{0, \ldots, 2^b - 1\}$. Thus, $x \in \{0,1\}^{n - \lceil \log_2(d) \rceil}$ represents a number $x \leq (2^{n - \log_2(d)} - 1) = 2^n/d - 1$, the product dx is at most $2^n - d$, and hence $dx + d - 1 \leq 2^n - 1$ can be written as an n-bit string.)

Theorem 3. *For $d \geq 1$, a random permutation $\pi \in \mathbb{F}_n$ and $q \leq 2^{n-1}/d^2$ is* $\textsc{twin}^d(x) = \bigoplus_{i=0}^{d-1} \pi(dx+i)$ *a (q,a)-secure PRF with*

$$a \leq \frac{qd^2}{2^n} + \frac{1}{2^{dn-1}} \sum_{0 \leq i < q} i^d.$$

Proof: As in the proof of Theorem 2, the adversary asks q queries $x_1, \ldots, x_q$, w.l.o.g. $x_i \neq x_j$ for $i \neq j$, and learns q responses $y_i = \textsc{twin}(x_i)$. We define $\pi_{di-d+1} = \pi(dx_i), \ldots, \pi_{di} = \pi(dx_i + d - 1)$, $N = \{0,1\}^n$, and

$$C^* = \{\, (s_1, \ldots, s_d) \in N^d | \exists (i,j) : i \neq j, s_i = s_j \,\}.$$

Clearly, the d-tuples $(\pi_{di-d+1}, \ldots, \pi_{di})$ are not in C^*. Note that $|C^*| = 2^{(d-1)n} * d(d-1)/2 \leq 2^{(d-1)n} d^2/2$. Similar to Equation (1), we define a set T_i^* of the d-tuples still avaliable: $T_i^* = T_i^{**} - C^*$, $T_1^{**} = N^d$, and

$$\begin{array}{rcl} T_{i+1}^{**} & = & N^d - \quad (\{\pi_1, \ldots, \pi_{di}\} \times N^{d-1}) \\ & & \quad - (N \times \{\pi_1, \ldots, \pi_{di}\} \times N^{d-2}) \\ & & \quad - \qquad\qquad \cdots \\ & & \quad - \quad (N^{d-1} \times \{\pi_1, \ldots, \pi_{di}\}). \end{array} \tag{5}$$

Note that $|T_{i+1}^{**}| \geq 2^{dn} - (d^2 * i * 2^{(d-1)n})$. We simulate generating the y_i:

For $i := 1$ to q: choose $(\pi_{di}, \ldots, \pi_{di+d-1}) \in_{\mathrm{R}} T_i^*$
output $y_i = \pi_{di} \oplus \cdots \oplus \pi_{di+d-1}$.

The sets T_{i+1}^{**} are $(di)^d$-fair:

Compare Equations (1) and (5). Set $j := di + 1$ and show the $(j-1)^d$-fairness of the sets T_j^{**} as in the proof of Theorem 2.

Choosing the d-tuples $(\pi_{di}, \ldots, \pi_{di+d-1})$ from fair sets:

The sets T_i^{**} are $(di - d)$-fair, i.e., c-sets or o-sets of size $(di-d)^d$ for T_i^{**} exist. We argue as in the proof of Theorem 2: If d is odd, then a c-set V_i for T_i^{**} exists. If d is even, an o-set U_i exists. Figure 2 describes the corresponding simulations.

In addition to Figure 1, the simulation in Figure 2 takes care that t_i is in T_i^*, not just in T_i^{**}. If the last output is *bad* $= 0$, all d-tuples t_i used to generate the y_i are uniformly chosen values from fair sets $T_i^{**} - U_i$ or $V_i \cup T_i^{**}$, hence $a \leq \mathrm{pr}[\mathrm{bad} = 1]$.

Evaluating pr[bad = 1]:

We get bad = 1 if and only if one of the two then-clauses is executed at least once. By B_i^1 we denote the event that the then-clause marked by $(*)$ is executed in round i, B^1 denotes the event that this clause is executed in any round $i \in \{1, \ldots, q\}$, i.e., $\mathrm{pr}[B^1] \leq \sum_{i=1}^{q} B_i^1$. For the then-clause marked by $(**)$, we define the similar events B_i^2 and B^2. Thus $\mathrm{pr}[\mathrm{bad} = 1] \leq \mathrm{pr}[B^1] + \mathrm{pr}[B^2]$. We start with $\mathrm{pr}[B^1]$:

$$\mathrm{pr}[B_i^1] = \left\{ \begin{array}{ll} (di-d)^d / |T_i^{**}| & \text{for even } d \\ (di-d)^d / (|T_i^{**}| + (di-d)^d) & \text{for odd } d \end{array} \right\} \leq \frac{(di-d)^d}{|T_i^{**}|}.$$

Even d:

```
Set bad := 0;
for $i$ := 1 to $q$:
    determine $T_i^{**}$ and $U_i$;
    $t_i = (\pi_{di-d+1}, \dots, \pi_{di}) \in_{\mathsf{R}} T_i^{**}$;
    if $t_i \in U_i$ then (*)
      bad := 1;
    if $t \in C^*$ then (**)
      $t_i \in_{\mathsf{R}} T_i^{**} \cap \overline{C^*}$;
    output $y_i := \pi_{di-d+1} \oplus \cdots \oplus \pi_{di}$;
output bad.
```

Odd d:

```
Set bad := 0;
for $i$ := 1 to $q$:
    determine $T_i^{**}$ and $V_i$;
    $t_i = (\pi_{di-d+1}, \dots, \pi_{di}) \in_{\mathsf{R}} T_i^{**} \cup V_i$;
    if $t \not\in T_i^{**}$ then (*)
      bad := 1; choose $t_i \in_{\mathsf{R}} T_i^{**}$;
    if $t \in C^*$ then (**)
      $t_i \in_{\mathsf{R}} T_i^{**} \cap \overline{C^*}$;
    output $y_i := \pi_{di-d+1} \oplus \cdots \oplus \pi_{di}$;
output bad.
```

Fig. 2. Two simulations for TWIN^d

Since $|T_{i+1}^{**}| \geq 2^{dn} - (d^2 * i * 2^{(d-1)n})$ and $i \leq q \leq 2^{n-1}/d^2$, we get $|T_{i+1}^{**}| \geq 2^{dn} - 2^{dn-1} \geq 2^{dn-1}$ and thus

$$\frac{(i-1)^d}{|T_i|} \leq \frac{(i-1)^d}{2^{dn-1}} \quad \Rightarrow \quad \mathrm{pr}[B^1] \leq \sum_{1 \leq i \leq q} \frac{(i-1)^d}{2^{dn-1}} = \frac{1}{2^{dn-1}} \sum_{0 \leq i < q} i^d.$$

Now we bound $\mathrm{pr}[B^2]$. Since $|T_i^{**}| \geq 2^{nd} - (d^2) * i * 2^{(d-1)n} \geq 2^{dn-1}$ for $i \leq q \leq 2^{n-1}/d^2$ and $|C^*| \leq 2^{(d-1)n} * d^2/2$ we get

$$\begin{aligned} \mathrm{pr}[B_i^2] &\leq \frac{|C^*|}{|T_i^{**}|} \leq \frac{2^{(d-1)n} * \frac{d^2}{2}}{2^{dn-1}} = \frac{2^{(d-1)n} * d^2}{2^{dn}} = \frac{d^2}{2^n} \\ \Rightarrow \quad \mathrm{pr}[B^2] &\leq \sum_{1 \leq i \leq q} \mathrm{pr}[B_i^2] \leq \sum_{1 \leq i \leq q} \frac{d^2}{2^n} = \frac{q * d^2}{2^n} \end{aligned}$$

hence $a \leq \mathrm{pr}[\mathrm{bad} = 1] \leq 2^{-dn+1} \sum_{0 \leq i < q} i^d + q * d^2/2^n$. □
Consider $d \in \{1, 2, 3\}$. Based on Equations (2)–(4), we get

$$\begin{aligned} d = 1&: \ a \leq \ \ q/2^n + q^2/2^n \\ d = 2&: \ a \leq 4q/2^n + q^3/(3 * 2^{2n-1}) \\ d = 3&: \ a \leq 9q/2^n + q^4/2^{3n+1}. \end{aligned}$$

The $(2^{-dn+1} \sum_{0 \leq i < q} i^d)$-term determines the maximum size of q, at least if d is such small and for practically interesting $n \geq 64$. We conclude: for small d, the PRF-security of TWIN^d is close to the PRF-security of SUM^d.

6 Final Comments

6.1 Practical Security

We presented constructions for PRFs from permutations, and we proved our PRFs to be (q, a)-secure if the permutations are $(\infty, 0)$-secure (or "ideal") PRPs.

In practice, our PRPs (i.e. block ciphers) are not ideal ones. What we actually are interested in is a close relationship between the derivation of the underlying permutations from being ideal $(\infty, 0)$-secure PRPs, and the derivation of the constructed PRF from being $(\infty, 0)$-secure. This is quite straightforward, and we exemplify this for the $\textsc{twin}^d$-construction:

Theorem 4. *Let q and a be chosen such that $\textsc{twin}^d$ is (q, a)-secure in the ideal case. Let B be a (t, qd, δ)-secure PRP. The function $f : \{0,1\}^{n-\lceil d \rceil} \to \{0,1\}^n$ defined by*

$$f(x) = B(dx) \oplus \cdots \oplus B(dx + d - 1) \tag{6}$$

is $(t - qt', q, a + \delta)$-secure. Here, t', denotes the time to evaluate Expression (6).

Note that the function f is indeed an instantiation of the PRF $\textsc{twin}^d$ using the concrete (non-ideal) PRP B.

Proof: [of Theorem 4] Assume an adversary A_f running at most $t - qt'$ units of time, asking for q values $f(x_1), \ldots f(x_q)$, achieves an advantage $\mathrm{Adv}^{\mathrm{Fun}}_{A_f, f} > a + \delta$. We describe an adversary A_B for B, using A_f as some kind of "subroutine". The performance of A_B disproves the (t, qd, δ)-security of B.

Whenever A_f chooses $x \in \{0,1\}^{n-\lceil d \rceil}$ and asks for $f(x)$, A_B asks for the values $B(dx)$, ..., $B(dx + d - 1)$ and evaluates Expression (6). A_B uses the output-bit produced by A_f as its own output-bit.

Running A_B requires the running time for A_f plus the additional time qt' for q evaluations of (6), and q queries for the f-oracle are translated into dq queries for the B-oracle. Since $\textsc{twin}^d$ is (q, a)-secure in the ideal case, and since A_f is assumed to achieve an advantage of more than $a + \delta$, the advantage of A_B in distinguishing between B and an ideal block cipher exceeds δ. □

Given an estimate of the number q of plaintext/ciphertext pairs the adversary can learn, and given the block size n, the security architect must decide on the size of the parameter d. Our analysis provides precise bounds (instead of asymptotic estimates) to help her making a reasonable decision. This kind of reasoning, the "concrete security analysis", was initiated in [4].

6.2 Super Pseudorandom Permutations

Luby and Rackoff [9] introduced a distinction between *super* PRPs and (ordinary) PRPs: For ordinary PRPs, the adversary may only choose values x and ask the oracle Q for $Q(x)$. Such adversaries are "chosen plaintext" adversaries. On the other hand, super PRPs need to resist "combined chosen plaintext / chosen ciphertext" adversaries, i.e., adversaries also able to choose y and ask for $Q^{-1}(y)$. For our constructions we don't need super PRPs – ordinary PRPs are sufficient. This makes our results all the more meaningful.

6.3 Comparison and Conclusion

This paper deals with the construction of PRFs from PRPs. We propose two constructions, $\textsc{sum}^d : \{0,1\}^n \to \{0,1\}^n$ and $\textsc{twin}^d\{0,1\}^{n-\lceil \log_2(d) \rceil} \to \{0,1\}^n$, based on PRPs over $\{0,1\}^n$.

Our constructions preserve the security of the underlying PRP better than the truncate construction from [7] and are much more efficient than the order construction, also from [7].

The truncate construction from [7] is re-considered in [3], claiming an improved security analysis compared to [7]. Also, [3] deals with SUM^2 and TWIN^2 – the two-dimensional variants of the constructions we scrutinise here. In short, if the number q of oracle queries is $q \ll 2^n/O(n)$, both the SUM^2 and the TWIN^2 construction are claimed to be secure. (For TWIN^2, a short sketch of proof is given.) Note that the results in claimed in [3] are significantly better than the results provided in the current paper.

Now consider data dependent re-keying, (DDRK) [5]. If k is the key size of the underlying block cipher, the result on the security of DDRK [5, Theorem 5.2] requires $q \ll 2^{4k/5}$. In fact, that result depends on the assumption that exhaustively searching $2^{4k/5}$ keys is infeasible. If, say, $k = 80$, the effective key-length guaranteed by the result is only $4{*}80/5\,\text{bit} = 64\,\text{bit}$. This is a disadvantage, compared to our schemes. (Note though: [5] conjecture that the bound on q can be improved to $q \ll 2^{k(1-\epsilon)}$.) Depending on which block cipher is used and on hardware constraints, the very frequent key changes needed for DDRK can constitute another disadvantage.

For some applications, e.g. on low-end smartcards, even the effort to switch between only d fixed secret keys may be prohibitive. In this case, the TWIN^d construction is superior to SUM^d, if a PRF with only $n - \lceil \log_2(d) \rceil$ input bits is acceptable.

Acknowledgements

Thanks to the anonymous referees for their helpful comments.

References

1. W. Aiello, R. Venkatesan, “Foiling Birthday Attacks in Length Doubling Transformations”, Eurocrypt 96, 307–320, Springer LNCS 1070.
2. M. Bellare, A. Desai, E. Jokipii, P. Rogaway, “A Concrete Security Treatment of Symmetric Encryption: The DES Modes of Operation”, FoCS 97, IEEE press.[3]
3. M. Bellare, R. Impagliazzo, “A Tool for Obtaining Tighter Security Analyses of Pseudorandom Function Based Constructions, with Applications to PRP→PRF Conversion” (FOCS ’99 submission), *Theory of Cryptography Library*[4], record 99-24 (1999).
4. M. Bellare, J. Kilian, P. Rogaway, “The Security of the Cipher Block Chaining Message Authentication Code”, Crypto 94 Proceedings, Springer LNCS 839.[5]
5. M. Bellare, T. Krovetz, P. Rogaway, “Luby-Rackoff Backwards: Increasing Security by Making Block Ciphers Non-Invertible”, Eurocrypt 98, Springer LNCS 1403.[6]

[3] Full version online: `http://wwwcsif.cs.ucdavis.edu/~rogaway/papers/list.html`

[4] The library is online: `http://philby.ucsd.edu/cryptolib/`

[5] Full version online: `http://www-cse.ucsd.edu/users/mihir/crypto-papers.html`

[6] Full version online: `http://www-cse.ucsd.edu/users/mihir/crypto-papers.html`

6. O. Goldreich, S. Goldwasser, S. Micali, "How to Construct Random Functions", Journal of the ACM, Vol. 33, No 4, 792–807 (1986).
7. C. Hall, D. Wagner, J. Kelsey, B. Schneier, "Building PRFs from PRPs", Crypto 98, Springer LNCS 1462.[7]
8. L. Levin, "One Way Functions and Pseudorandom Generators", *Combinatorica*, 7 (4), 357–363 (1987).
9. M. Luby, C. Rackoff, "How to Construct Pseudorandom Permutations from Pseudorandom Functions", *SIAM J. Comput.*, Vol. 17, No. 2, 373–386, (1988).
10. S. Lucks, "Faster Luby-Rackoff Ciphers", Fast Software Encryption 1996, Springer LNCS 1039, 189–205.
11. M. Naor, O. Reingold, "On the Construction of Pseudo-Random Permutations: Luby-Rackoff Revisited", *J. of Cryptology*, Vol. 12, No. 1, 29–66 (1999).
12. J. Patarin, "About Feistel Schemes with Six (or More) Rounds", Fast Software Encryption 1997, Springer LNCS 1372, 103–121.

Appendix: The 2-Dimensional Construction $\mathrm{SUM}^2(x) = \sigma(x) \oplus \tau(x)$

To improve the tangibility of this paper, the abstract deals with a simple but non-trivial special case of $\textsc{sum}^d$, the 2-dimensional variant

$$\textsc{sum}^2(x) = \sigma(x) \oplus \tau(x),$$

depending on two permutations $\sigma, \tau\{0,1\}^n \to \{0,1\}^n$. Not surprisingly, $\textsc{sum}^2$ is not a $(\infty, 0)$-secure PRF. In fact, collisions are too probable. E.g., the probability that the first two pseudorandom values y_1 and y_2 generated by using $\textsc{sum}^2$ to collide is too high: $\mathrm{pr}[y_1 = y_2] > 2^{-n}$. To see this, consider simulating $\textsc{sum}^2$.

Initially, there are 2^{2n} pairs $(s,t) \in \{0,1\}^n$ to choose for $(\sigma(x_1), \tau(x_1))$. For every value $y \in \{0,1\}^n$, there exist exactly 2^n pairs (s,t) with $\sigma(x_1) \oplus \tau(x_1) = y_1$.

Let $x_2 \neq x_1$. In the second step, a pair $(s', t') = (\sigma(x_2), \tau(x_2))$ is chosen with $s' \neq s$ and $t' \neq t$. There are 2^{n-1} values $s' \neq s$ and as much values $t' \neq t$, hence the number of such pairs is $(2^{n-1})^2$. For every value $s' \neq s$, exactly one value $t' \neq t$ exists with $s' \oplus t' = s \oplus t$, and $y_1 = y_2$ if and only if $s' \oplus t' = s \oplus t$. Hence, exactly $2^n - 1$ of the $(2^{n-1})^2$ possible pairs (s', t') induce $y_2 = y_2$, and thus

$$\mathrm{pr}[y_1 = y_2] = \frac{2^n - 1}{(2^n - 1)^2} = \frac{1}{2^n - 1}.$$

If $\textsc{sum}^2$ where an ideal random function, we had $\mathrm{pr}[y_1 = y_2] = 2^{-n}$. But how good is the PRF $\textsc{sum}^2$ actually?

Theorem 5. *For random permutations $\sigma, \tau \in \mathbb{F}_n$ and $q \leq 2^{n-1}$, the function f with $f(x) = \textsc{sum}^2(x) = \sigma(x) \oplus \tau(x)$ is a (q, a)-secure PRF with $a = q^3/2^{2n-1}$.*

[7] Full version online: `http://www.counterpane.com/publish-1998.html`

Proof: The set $T \subseteq (\{0,1\}^n)^2$ is "fair",[8] if for every value $y \in \{0,1\}^n$

$$\left|\{\, (\sigma_*, \tau_*) \in T \mid \sigma_* \oplus \tau_* = y \,\}\right| = \frac{|T|}{2^n}.$$

The adversary A asks $q \leq 2^{n-1}$ oracle queries $x_1, \ldots, x_q$, w. l. o. g. $x_i \neq x_j$ for $i \neq j$. We write $y_1, \ldots, y_q$ for the corresponding oracle responses.

Consider evaluating SUM^2 by choosing a pair $(\sigma_i, \tau_i) = (\sigma(x_i), \tau(x_i))$ and computing $y_i = \sigma_i \oplus \tau_i$. If all (σ_i, τ_i) where randomly chosen from a fair set and uniformly distributed, then the sums $y_i = \sigma_i \oplus \tau_i$ would be uniformly distributed random values – un-distinguishable from the output of a random function.

The remainder of this proof is organised as follows:

1. We describe the sets $T_i \subseteq (\{0,1\}^n)^2$ the pairs (σ_i, τ_i) are chosen from, and we specify fair subsets $U_i \subseteq T_i$ with $|U_i| = |T_i| - (i-1)^2$.
2. We describe how to choose the pairs (σ_i, τ_i) from the fair sets U_i, except when a "bad" event happens.
3. We calculate the probability of the "bad" event.

Let $i \neq j$. Since σ and τ are permutations, $\sigma_i \neq \sigma_j$ and $\tau_i \neq \tau_j$. Thus, by choosing the pair (σ_i, τ_i), all pairs (s, t) with $s = \sigma_i$ or $t = \tau_i$ are "consumed", i.e., cannot be chosen for (σ_j, τ_j).

By S_i, we denote the set of consumed pairs before the choice of (σ_i, τ_i). By $T_i = (\{0,1\}^n)^2 - S_i$, we denote the set of un-consumed pairs. Note that (T_i is fair) $\Leftrightarrow$ (S_i is fair). Since $S_1 = \{\}$, both S_1 and T_1 are fair and y_1 is a uniformly distributed random value. Given $(\sigma_1, \tau_1), \ldots, (\sigma_k, \tau_k)$ we define $U_{k+1} \subseteq T_{k+1}$. Consider the following $2k$ fair sets of pairs:

$$\begin{gathered}\{(\sigma_1, \tau_*) \mid \tau_* \in \{0,1\}^n\}, \ldots, \{(\sigma_k, \tau_*) \mid \tau_* \in \{0,1\}^n\} \\ \text{and } \{(\sigma_*, \tau_1) \mid \sigma_* \in \{0,1\}^n\}, \ldots, \{(\sigma_*, \tau_k) \mid \sigma_* \in \{0,1\}^n\}.\end{gathered}$$

S_{k+1} is the union of the above $2k$ sets of pairs. If the above 2k sets were all disjoint, S_{k+1} would be fair. But actually, exactly k^2 pairs are contained in two of the above sets, namely all pairs (σ_i, τ_j) with $i, j \in \{1, \ldots, k\}$. We arbitrarily choose k^2 unique representatives (σ'_i, τ'_j) for (σ_i, τ_j) with $(\sigma'_i, \tau'_j) \in T_{k+1}$ and $\sigma'_i \oplus \tau'_j = \sigma'_i \oplus \tau'_j$. We define U_{k+1} to be the set of all pairs in T_{k+1} except for the representatives (σ'_i, τ'_j). Hence $|U_{k+1}| = |T_{k+1}| - k^2$. By induction one can see that for every $y \in \{0,1\}^n$ the set U_{k+1} contains exactly $2^n - 2k$ pairs (σ_s, τ_t) with $\sigma_s \oplus \tau_t = y$. Since $k \leq q \leq 2^{n-1}$, it is possible run the simulation described in Figure 3, especially, a set U_i exists.

The distribution of the values y_i is as required for SUM^2. The simulation generates an additional value "bad". If bad $= 0$, each of the pairs (σ_i, τ_i) is

[8] This notion of fairness is the two-dimensional special case of Definition 2. If T is fair and we choose $(s, t) \in_{\mathsf{R}} T$, the sum $y = s \oplus t$ is a uniformly distributed random value in $\{0,1\}^n$.

```
Set bad := 0;
for $i := 1$ to $q$: determine the sets $T_i$ and $U_i$;
                choose $(\sigma_i, \tau_i) \in_{\mathsf{R}} T_i$;
                if $(\sigma_i, \tau_i) \notin U_i$ then bad := 1;
                output $y_i = \sigma_i \oplus \tau_i$;
output bad.
```

Fig. 3. A simulation for the PRF $\textsc{sum}^2$

chosen from a fair set U_i, and the sums y_i are un-distinguishable from the output of a random function. Thus $\mathrm{Adv}^{\mathrm{Fun}}_{A,f} \leq \mathrm{pr}[\mathrm{bad} = 1]$ for every adversary A. Using

$$\mathrm{pr}[(\sigma_{i+1}, \tau_{i+1}) \notin U_{i+1}] = \frac{|T_{i+1}| - |U_{i+1}|}{|T_{i+1}|} = \frac{i^2}{(2^n - i)^2},$$

we bound the probability $\mathrm{pr}[\mathrm{bad} = 1]$:

$$\begin{aligned} \mathrm{pr}[\mathrm{bad} = 1] \quad &\leq \sum_{1 \leq i \leq q} \mathrm{pr}[(\sigma_i, \tau_i) \notin U_i] \quad = \sum_{0 \leq i < q} \frac{i^2}{(2^n - i)^2} \\ &\leq \sum_{0 \leq i < q} \frac{i^2}{(2^n - q)^2} \quad = \quad (2^n - q)^{-2} * \sum_{0 \leq i < q} i^2. \end{aligned}$$

Since $q \leq 2^{n-1}$

$$\mathrm{pr}[\mathrm{bad} = 1] \leq (2^{n-1})^{-2} * \sum_{0 \leq i < q} i^2. \qquad (7)$$

By using $\sum_{0 \leq i < q} i^2 = (q(q-1)(2q-1))/6 \leq 2q^3/6$ we get

$$\mathrm{pr}[\mathrm{bad} = 1] \leq \frac{q^3}{3 * (2^n - q)^2}$$

and hence $\mathrm{pr}[\mathrm{bad} = 1] \leq q^3/2^{2n-1}$. □

Note that Theorem 5 provides a marginally better bound than Theorem 2 for $d = 2$. This is, because the Theorem 2 considers the general case (and because the current author tried to avoid overcrowding its proof with too many technical details). The general outline of the proofs of Theorems 2, 3, and 5 is quite similar.