# Necessary and Sufficient Conditions for Collision-Free Hashing

Alexander Russell*
acr@theory.lcs.mit.edu

Laboratory for Computer Science
545 Technology Square
Massachusetts Institute of Technology
Cambridge, MA 02139 USA

**Abstract.** This paper determines an exact relationship between collision-free hash functions and other cryptographic primitives. Namely, it introduces a new concept, the pseudo-permutation, and shows that the existence of collision-free hash functions is equivalent to the existence of claw-free pairs of pseudo-permutations. When considered as one bit contractors (functions from $k+1$ bits to $k$ bits), the collision-free hash functions constructed are more efficient than those proposed originally, requiring a single (claw-free) function evaluation rather than $k$.

## 1 Introduction

Hash functions with various cryptographic properties have been studied extensively, especially with respect to signing algorithms (see [2, 3, 4, 10, 12, 14, 15]). We focus on the most natural of these functions, the *collision-free* hash functions. A hash function $h$ is collision-free if it is hard for any efficient algorithm, given $h$ and $1^k$, to find a pair $(x, y)$ so that $|x| = |y| = k$ and $h(x) = h(y)$. These functions were first carefully studied by Damgård in [2]. Given the interest in these functions, we would like to determine necessary and sufficient conditions for their existence in terms of other, simpler, cryptographic machinery.

There has been recent attention to the minimal logical requirements for other cryptographic primitives. Rompel (in [12]), improving a construction of Naor and Yung (in [10]), shows that the existence of secure digital signing systems (in the sense of [5]) is equivalent to the existence of one-way functions. Impagliazzo, Levin, and Luby (in [7]) and Håstad (in [6]) demonstrate the equivalence of the existence of pseudo-random number generators (see [1, 13]) and the existence of one-way functions.

Damgård (in [2]), distilling arguments of Goldwasser, Micali, and Rivest (in [5]), shows that the existence of another cryptographic primitive, a *claw-free* pair of permutations, is sufficient to construct collision-free hash functions. A pair of permutations $(f, g)$ is *claw-free* if it is hard for any efficient algorithm, given $(f, g)$ and $1^k$, to find a pair $(x, y)$ so that $|x| = |y| = k$ and $f(x) =$

* Supported by a NSF Graduate Fellowship

$g(y)$. Comparing the definitions of collision-free hash functions and claw-free pairs of permutations, it seems unlikely that the existence of claw-free pairs of permutations is necessary for the existence of collision-free hash functions because the hash functions have no explicit structural properties that reflect the condition of permutativity in the claw-free pairs of permutations. Our paper relaxes this condition of permutativity and defines a natural object, the existence of which is necessary and sufficient for the existence of a family of collision-free hash functions.

We define a new concept, the *pseudo-permutation.* A function $f : S \rightarrow S$ is a pseudo-permutation if it is computationally indistinguishable from a permutation. For this "indistinguishability" we require that it be hard for any efficient algorithm, given the function $f$ and $1^k$, to compute a quickly verifiable proof of non-injectivity, i.e. a pair $(x, y)$ where $|x| = |y| = k, x \neq y$, and $f(x) = f(y)$. The main contribution of our paper is that the existence of a collection of claw-free pairs of pseudo-permutations is equivalent to the existence of a collection of collision-free hash functions. This fact shows that nontrivial "claw-freeness" is essential to collision-free hashing and also weakens the assumptions necessary for their existence.

In §2 we describe our notation and define some cryptographic machinery. In §3 we present our main theorem. In §4 we consider the efficiency of our construction. Finally, in §5, we discuss an open problem and the motivation for this research.

## 2 Notation and Definitions

We adopt the following class of *expected polynomial time Turing machines* as our standard class of "efficient algorithms" (see [9] for a precise definition and discussion of this class).

**Definition 1.** Let $\mathcal{EA}$, our class of efficient algorithms, be the class of probabilistic Turing machines (with output) running in expected polynomial time. We consider these machines to compute probability distributions over $\Sigma^*$. For $M \in \mathcal{EA}$ we use the notation $M[w]$ to denote both the probability space defined by $M$ on $w$ over $\Sigma^*$ and an element selected according to this space.

For simplicity, let us fix a two letter alphabet $\Sigma = \{0, 1\}$. The consequences of a larger alphabet will be discussed in §4. $1^k$ denotes the concatenation of $k$ 1's. $\mathcal{Q}[x]$ denotes the class of polynomials over the rationals. Borrowing notation from [4], if $S$ is a probability space, $x \leftarrow S$ denotes the assignment of $x$ according to $S$. If $p(x_1, \ldots, x_k)$ is a predicate, then $\Pr[x_1 \leftarrow S_1, \ldots, x_k \leftarrow S_k : p(x_1, \ldots, x_k)]$ denotes the probability that $p$ will be true after the ordered assignment of $x_1$ through $x_k$.

**Definition 2. A collection of claw-free functions** is a collection of function tuples $\{(f_i^0, f_i^1) | i \in I\}$ for some index set $I \subseteq \Sigma^*$ where $f_i^j : \Sigma^{|i|} \rightarrow \Sigma^{|i|}$ and:

CF1. *[accessibility]* there exists a generating algorithm $G \in \mathcal{EA}$ so that $G[1^n] \in \{0,1\}^n \cap I$.

CF2. *[efficient computability]* there exists an computing algorithm $C \in \mathcal{EA}$ so that for $i \in I, j \in \{0,1\}$, and $x \in \Sigma^{|i|}, C[i,j,x] = f_i^j(x)$.

CF3. *[claw-freedom]* for all claw finding algorithms $A \in \mathcal{EA}$, $\forall P \in \mathcal{Q}[x]$, $\exists k_0$, $\forall k > k_0$,

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : f_i^0(x) = f_i^1(y)] < \frac{1}{P(k)}$$

If $(f^0, f^1)$ is a member of a collection of claw-free pairs, then $(f^0, f^1)$ is called a *claw-free pair* and a pair $(x,y)$ so that $f^0(x) = f^1(y)$ is called a *claw* of $(f^0, f^1)$.

This definition, from a cryptographic perspective, requires nothing of the function pairs involved unless they have overlapping images. One way to require that the functions have overlapping images is to require that the functions be permutations. This yields the following object, originally defined in [5] and then in this form by [2].

**Definition 3. A collection of claw-free permutations** is a collection of claw free functions $\{(f_i^0, f_i^1) | i \in I\}$ where each $f_i^j$ is a permutation.

Although the intractability of certain number theoretic problems implies the existence of a collection of claw-free pairs of permutations[2], the existence of one-way permutations is not known to be enough.[3]

**Definition 4. A collection of pseudo-permutations** is a collection of functions $\{f_i | i \in I\}$ for some index set $I \subseteq \Sigma^*$ where $f_i : \Sigma^{|i|} \rightarrow \Sigma^{|i|}$ and:

$\psi$P1. *[accessibility]* there exists a generating algorithm $G \in \mathcal{EA}$ so that $G[1^n] \in \{0,1\}^n \cap I$.

$\psi$P2. *[efficient computability]* there exists a computing algorithm $C \in \mathcal{EA}$ so that for $i \in I$ and $x \in \Sigma^{|i|}, C[i,x] = f_i(x)$.

$\psi$P3. *[collapse freedom]* for all collapse finding algorithms $A \in \mathcal{EA}$, $\forall P \in \mathcal{Q}[x]$, $\exists k_0$, $\forall k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : f_i(x) = f_i(y) \wedge x \neq y] < \frac{1}{P(k)}$$

If a function $f$ is a member of a collection of pseudo-permutations it is called a *pseudo-permutation* and a pair $(x,y)$ where $f(x) = f(y)$ and $x \neq y$ is called a *collapse* of $f$. Property $\psi$P3 means that it is hard for an efficient algorithm to produce a quickly verifiable proof that $f$ is not a permutation. In the definition above, this proof is specifically required to be a proof of non-injectivity: a collapse. One might also prove that a function $f : S \rightarrow S$ is

[2] In [5] the intractability of factoring is shown to be sufficient. In [2], the construction of [5] is extended and the intractability of the discrete log is also shown to be sufficient.

[3] [11] discusses algebraic forms of one way permutations sufficient for claw-free permutations.

not a permutation by producing a proof of non-surjectivity: an element in $S - \mathbf{Im} f$. We require the former because of the difference in computational resources necessary to verify these proofs: a proof of non-injectivity may be verified with two function applications whereas a proof of non-surjectivity requires evaluation of $f$ at every point in the domain. Like the definition for claw-free functions, the above definition requires nothing cryptographically of the functions involved unless $|\mathbf{Im}\, f_i| < |\mathbf{Dom}\, f_i|$. If the functions in the collection are injective, then $\psi$P3 is vacuously true.

Pseudo-permutations are a reasonable replacement for permutations in a cryptographic setting; for example, the entire signing algorithm of Naor and Yung (in [10]) may be implemented with one-way[4] pseudo-permutations rather than one-way permutations.

**Definition 5.** **A collection of claw-free pseudo-permutations** is a collection of claw-free functions $\{(f_i^0, f_i^1) | i \in I\}$ so that both $\{f_i^0 | i \in I\}$ and $\{f_i^1 | i \in I\}$ are collections of pseudo-permutations.

Collections of claw-free pseudo-permutations gather their cryptographic structure from the tension between two otherwise weak definitions. If the pseudo-permutations lack cryptographic richness (so that they are very close to permutations) then the intersection of their images must be large and there must be many claws, imparting richness by virtue of claw-freedom. If, instead, the pair has few claws, then the images of the two functions must be nearly disjoint (and so, small) so that the functions themselves are cryptographically rich by virtue of their many collapses.

## 3 The Structure of Collision-Free Hash Functions

**Definition 6.** **A collection of collision-free hash functions** is a collection of functions $\{h_i | i \in I\}$ for some index set $I \subseteq \Sigma^*$ where $h_i : \Sigma^{|i|+1} \to \Sigma^{|i|}$ and:

H1. *[accessibility]* there exists a generating algorithm $G \in \mathcal{EA}$ so that $G[1^n] \in \{0,1\}^n \cap I$.

H2. *[efficient computability]* there exists a computing algorithm $C \in \mathcal{EA}$ so that for $i \in I$, and $w \in \Sigma^{|i|+1}, C[i, w] = h_i(w)$.

H3. *[collision-freedom]* for all collision generating algorithms $A \in \mathcal{EA}, \forall P \in Q[x], \exists k_0, \forall k > k_0$

$$\Pr[i \leftarrow G[1^k], (x, y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y] < \frac{1}{P(k)}$$

If $h$ is a member of a collection of collision-free hash functions then $h$ is called a *collision-free hash function* and a pair $(x, y)$ where $h(x) = h(y)$ and $x \neq y$ is called a *collision* of $h$.

[4] This is a collection of pseudo-permutations which are hard to invert in the sense of one-way functions.

The notion of a polynomial separator will be used in the following proof. For the purposes of this paper, a separator is a pair of bijections from $\Sigma^k$ into $\Sigma^{k+1}$ so that their images have no intersection. (Because $|\Sigma| = 2$, their images cover $\Sigma^{k+1}$.)

**Definition 7. A collection of polynomial separators** is a collection of function pairs $\{(\sigma_i^0, \sigma_i^1) | i \in I\}$ for some index set $I \subseteq \Sigma^*$ where $\sigma_i^j : \Sigma^{|i|} \to \Sigma^{|i|+1}$ for $j \in \{0,1\}$ and:

PS1. *[accessibility]* there exists a generating algorithm $G \in \mathcal{EA}$ so that $G[1^n] \in \{0,1\}^n \cap I$.
PS2. *[injectivity]* $\sigma_i^0$ and $\sigma_i^1$ are injective.
PS3. *[disjointness]* $\mathbf{im}\, \sigma_i^0 \cap \mathbf{im}\, \sigma_i^1 = \emptyset$
PS4. *[efficient computability]* there exists a computing algorithm $C \in \mathcal{EA}$ so that for $i \in I, w \in \Sigma^{|i|}$, and $j \in \{0,1\}, C[i,j,w] = \sigma_i^j(w)$.

With each such collection, we associate a collection of inverses $\{\iota_i\}$ where $\iota_i : \Sigma^{|i|+1} \to \Sigma^{|i|}$ and $\iota_i \circ \sigma_i^0 = \iota_i \circ \sigma_i^1 = \mathbf{id}_{\Sigma^{|i|}}$ and a collection of image deciders $\{\delta_i\}$ where $\delta_i : \Sigma^{|i|+1} \to \{0,1\}$ and $\forall w \in \Sigma^{|i|+1}, w \in \mathbf{im}\, \sigma_i^{\delta_i(w)}$.

The collection is said to have a **polynomial inverse** if the collection of inverses is so that $\exists C_\iota \in \mathcal{EA}, \forall w \in \Sigma^{|i|+1}, \forall i \in I, C_\iota[w,i] = \iota_i(w)$. If a collection is so endowed, then it is clear that the image deciders are also efficiently computable.

Construction of a family of polynomial separators with a polynomial inverse is easy: the *append*$_0$ : $x \mapsto x0$ and *append*$_1$ : $x \mapsto x1$ functions, for example.

**Theorem 8.** *There exists a collection of collision-free hash functions iff there exists a collection of claw-free pairs of pseudo-permutations.*

*Proof.* ($\Rightarrow$) Let $\{h_i | i \in I\}$ be a collection of collision-free hash functions and let $\{(\sigma_i^0, \sigma_i^1) | i \in I\}$ be a collection of polynomial separators (unrelated to the hash functions, but over the same index set). Define the collection $\{(f_i^0, f_i^1) | i \in I\}$ so that

$$f_i^j = h_i \circ \sigma_i^j \text{ for } j \in \{0,1\}$$

We show that the collection of functions so defined is a collection of claw-free pseudo-permutations. Properties *CF1* and *CF2* are immediate. Assume that property *CF3* does not hold, that is $\exists A \in \mathcal{EA}, \exists P \in \mathcal{Q}[x], \forall k_0, \exists k > k_0$,

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : f_i^0(x) = f_i^1(y)] \geq \frac{1}{P(k)}$$

Let $(x,y)$ be a claw for $(f_i^0, f_i^1)$, then $f_i^0(x) = f_i^1(y) \Rightarrow h_i(\sigma_i^0(x)) = h_i(\sigma_i^1(y))$, but $\mathbf{im}\, \sigma_i^0 \cap \mathbf{im}\, \sigma_i^1 = \emptyset$ so that $\sigma_i^0(x) \neq \sigma_i^1(y)$ and a collision has been found for $h_i$. Then, given this claw generating algorithm A we can construct a collision generating algorithm $A'$ succeeding with identical probability as $A$, violating *H3*. Therefore, *CF3* holds.

To show that $\{f_i^j | i \in I\}$ for each $j \in \{0,1\}$ are collections of pseudo-permutations, we verify properties $\psi$P1 – $\psi$P3 for each. $\psi$P1 and $\psi$P2 are immediate. Suppose, for contradiction, that property $\psi$P3 is not satisfied, so that $(\exists j \in \{0,1\},)\ \exists A \in \mathcal{EA}, \exists P \in \mathcal{Q}[x], \forall k_0, \exists k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : f_i^j(x) = f_i^j(y)] \geq \frac{1}{P(k)}$$

Let $(x,y)$ be a collapse of $f_i^j$, so that $f_i^j(x) = f_i^j(y)$ and $x \neq y$. Then $\sigma_i^j(x) \neq \sigma_i^j(y)$ because $\sigma_i^j$ is injective, so that $(\sigma_i^j(x), \sigma_i^j(y))$ is a nontrivial collision of $h_i$ (because $f_i^j = h_i \circ \sigma_i^j$). Then, given this collapse generating algorithm A we can construct a collision generating algorithm $A'$ succeeding with identical probability as $A$, violating *H3*. Therefore, *$\psi$P3* holds.

($\Leftarrow$) Let $\{(f_i^0, f_i^1) | i \in I\}$ be a collection of claw-free pairs of pseudo-permutations and let $\{(\sigma_i^0, \sigma_i^1) | i \in I\}$ be a collection of polynomial separators with inverses $\{\iota_i | i \in I\}$ and image deciders $\{\delta_i | i \in I\}$. Then define $\{h_i | i \in I\}$ so that

$$h_i(x) = f_i^{\delta_i(x)}(\iota_i(x))$$

We show that $\{h_i | i \in I\}$ is a collection of collision-free hash functions. Properties *H1* and *H2* are immediate. Assume, for contradiction, that property *H3* is not satisfied, that is $\exists A \in \mathcal{EA}, \exists P \in \mathcal{Q}[x], \forall k_0, \exists k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y] \geq \frac{1}{P(k)}$$

so that $\forall k_0, \exists k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y \wedge \delta_i(x) = \delta_i(y)] \geq \frac{1}{2P(k)} \bigvee$$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y \wedge \delta_i(x) \neq \delta_i(y)] \geq \frac{1}{2P(k)}$$

and we encounter at least one of two possibilities:

1. $\forall k_0, \exists k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y \wedge \delta_i(x) = \delta_i(y)] \geq \frac{1}{2P(k)}$$

2. $\forall k_0, \exists k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y \wedge \delta_i(x) \neq \delta_i(y)] \geq \frac{1}{2P(k)}$$

In the event of 1 above, the algorithm $A$ generates collisions $(x, y)$ where $\delta_i(x) = \delta_i(x)$. In this case, for at least one $j \in \{0,1\}, \forall k_0, \exists k > k_0$

$$\Pr[i \leftarrow G[1^k], (x,y) \leftarrow A[i] : h_i(x) = h_i(y) \wedge x \neq y \wedge j = \delta_i(x) = \delta_i(y)] \geq \frac{1}{4P(k)}$$

Given a collision of this sort, $h_i(x) = h_i(y) \Rightarrow f_i^j(x) = f_i^j(y)$ which is a collapse of $f_i^j$. Then, given algorithm $A$, we may produce another algorithm $A'$ which produces a collapse of $f_i^j$ with success related to the success of $A$ by a constant, violating *ψP3*.

In the event of 2 above, the algorithm $A$ generates collisions $(x, y)$ where $\delta_i(y) \neq \delta_i(x)$. A collision of this sort produces a claw because $h_i(x) = h_i(y) \Rightarrow f_i^{\delta_i(x)}(\iota_i(x)) = f_i^{\delta_i(y)}(\iota_i(y))$. Then, with algorithm $A$, we may construct a claw generating algorithm $A'$ which produces claws with success related to the success of $A$ by a constant, violating *CF3*.

A pair of separators partitions $\Sigma^{k+1}$ into two equal sized subsets (the images of the separators). We couple the definition of collision-free hash functions with the definition of polynomial separators to define a class of hash functions where every collision occurs across the partition boundary — then $h(x) = h(y)$ implies that $x$ and $y$ are in the images of different separators.

**Definition 9. A collection of separated collision-free hash functions** is a collection of function tuples $\{(h_i, \sigma_i^0, \sigma_i^1) | i \in I\}$ so that $\{h_i | i \in I\}$ forms a collection of collision-free hash functions, $\{(\sigma_i^0, \sigma_i^1) | i \in I\}$ forms a collection of polynomial separators, and

SH. *[separation]* $\forall j \in \{0,1\}$, $h_i|_{\mathbf{im}\,\sigma_i^j}$, the restriction of $h_i$ to $\mathbf{im}\,\sigma_i^j$, is bijective. Equivalently, $h_i(x) = h_i(y) \Rightarrow \delta_i(x) \neq \delta_i(y)$, where $\{\delta_i | i \in I\}$ is the collection of image deciders for the separators.

The existence of a collection of separated collision-free hash functions is equivalent to the existence of a collection of claw-free pairs of permutations.

**Theorem 10.** *There exists a collection of claw-free permutations iff there exists a collection of separated collision-free hash functions.*

*Proof.* This proof is omitted due its similarity with the previous proof.

The collision-free hash functions constructed in the two theorems above naturally inherit properties from the primitives with which they are constructed. If, for example, the claw-free pairs of (pseudo-) permutations are *trapdoor* functions, then the hash functions constructed share this property. It is not clear that the original hash functions constructed in [2] offer inheritance of this sort.

Toshiya Itoh [8] has pointed out that in the above constructions, the demand of claw-freedom can be replaced in an appropriate way with the demand of "distinction intractibility" as discussed in [14].

It is not hard to show that by composition the above collections of hash functions can be used to construct families of collision-free hash functions $\{h_i : i \in I\}$ where $h_i : \Sigma^{P(|i|)} \rightarrow \Sigma^{|i|}$ for any polynomial $P \in \mathcal{Q}[x]$ where $\forall x \in \mathcal{N}, P(x) > x$.

## 4 Comments on Efficiency

The ($\Leftarrow$) part of theorem 8 constructs a family of collision-free hash functions which are one bit contractors (functions from $\Sigma^{k+1}$ to $\Sigma^k$) and require 1 claw-free function evaluation to compute. Building a family of contractors by applying the construction in [2] yields hash functions which require $k$ evaluations of the underlying claw-free functions. For the case of one-bit contractors, then, the above construction is substantially more efficient.

In general, to construct hash functions from $\Sigma^{P(k)}$ to $\Sigma^k$ (for a polynomial $P$) one can do better than naive composition. Using arguments similar to those of Damgård in [2], the construction above can be altered to yield hash functions from $\Sigma^{P(k)}$ to $\Sigma^k$ which require $P(k) - k$ evaluations of the underlying claw-free functions on $k$ bit arguments. The collection constructed in [2] of the same sort requires $P(k)$ evaluations, so the efficiency improvement in this case is only an additive factor of $k$.

In [2], Damgård shows that expanding the size of the alphabet (and using claw-free *tuples* of functions) can reduce the number of required claw-free function evaluations by a multiplicative constant factor. This same procedure is applicable to our above construction.

## 5 An Open Problem

The motivation for this research is the following open problem: Is the existence of one-way functions sufficient for the existence of collision-free hash functions? We believe this to be the case, and that this paper represents a step towards proving this goal by demonstrating the equivalence between collision-free hashing and a primitive not requiring pure cryptographic permutativity.

## 6 Acknowledgements

We gratefully acknowledge the keen guidance of Silvio Micali, who originally suggested this problem. We also acknowledge Ravi Sundaram for several helpful discussions.

## References

1. Manual Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal of Computing*, 13(4):850–864, November 1984.

2. Ivan Damgård. Collision free hash functions and public key signature schemes. In *Proceedings of EUROCRYPT '87*, volume 304 of *Lecture Notes in Computer Science*, pages 203–216, Berlin, 1988. Springer-Verlag.
3. Alfredo De Santis and Moti Yung. On the design of provably-secure cryptographic hash functions. In *Proceedings of EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 412 – 431, Berlin, 1990. Springer-Verlag.
4. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the Association for Computing Machinery*, 33(4):792–807, October 1986.
5. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attack. *SIAM Journal of Computing*, 17(2):281–308, April 1988.
6. J. Håstad. Pseudo-random generators under uniform assumptions. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 395–404. ACM, 1990.
7. Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 12–24. ACM, 1989.
8. Toshiya Itoh. Personal comminucation, August 1992.
9. Leonid A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15:285–286, 1986.
10. M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 33–43. ACM, 1989.
11. Wakaha Ogata and Kaoru Kurosawa. On claw free families. In *Proceedings of ASIACRYPT '91*, 1991.
12. John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, pages 387–394. ACM, 1990.
13. A. Yao. Theory and applications of trapdoor functions. In *Proceedings of the Twenty Third IEEE Symposium on Foundations of Computer Science*, pages 80–91. IEEE, 1982.
14. Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. Duality between two cryptographic primitives. In *Proceedings of the Eighth International Conference on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 508 of *Lecture Notes in Computer Science*, pages 379–390, Berlin, 1990. Springer-Verlag.
15. Yuliang Zheng, Tsutomu Matsumoto, and Hideki Imai. Structural properties of one-way hash functions. In *Proceedings of CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 285–302, Berlin, 1990. Springer-Verlag.