Reprinted from

Information Processing Letters

Information Processing Letters 64 (1997) 239–244

An observation on associative one-way functions in complexity theory

Muhammad Rabi[1], Alan T. Sherman[*,2]

Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD 21250, USA

Received 1 August 1997
Communicated by S.G. Akl

ELSEVIER

ELSEVIER

Information Processing Letters 64 (1997) 239–244

Information Processing Letters

An observation on associative one-way functions in complexity theory

Muhammad Rabi [1], Alan T. Sherman [*,2]

Department of Computer Science and Electrical Engineering, University of Maryland Baltimore County, Baltimore, MD 21250, USA

Received 1 August 1997
Communicated by S.G. Akl

Abstract

We introduce the notion of associative one-way functions and prove that they exist if and only if P ≠ NP. As evidence of their utility, we present two novel protocols that apply strong forms of these functions to achieve secret-key agreement and digital signatures. © 1997 Published by Elsevier Science B.V.

Keywords: Associative one-way functions (AOWFs); Computational complexity; Cryptology; Cryptography; Cryptocomplexity; Cryptographic protocols; Digital signatures; Key-agreement problem; Secret-key exchange; Theory of computation; Public-key cryptography

1. Introduction

Two fundamental properties from algebra and cryptography are associativity (of function application) and one-wayness (of a cryptographic function). By construction, we prove that functions that satisfy both of these properties exist if and only if P ≠ NP.

Let $\mathcal{M}$ be any message space; for example, a practitioner might choose $\mathcal{M} = \{0,1\}^n$ for some positive integer n. An *associative one-way function* (AOWF) on $\mathcal{M}$ is any binary function $\circ : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ that is both associative and one-way. Function $\circ$ is associative if, for all $x, y, z \in \mathcal{M}$, $x \circ (y \circ z) = (x \circ y) \circ z$. Intuitively, $\circ$ is one-way if $\circ$ is easy to compute but hard to invert. Throughout we work in a worst-case complexity-theoretic framework for studying one-way functions.

The idea of associative one-way functions is due to Sherman, who proposed the concept in 1984 in his exploration of relationships among algebraic and security properties of cryptographic functions [18,10]. Since worst-case models have little to do with practical cryptology, our existence proof is mainly of interest in complexity theory. Nevertheless, we expect that the intriguing AOWF concept will be shown to have many useful applications in cryptography, possibly in the key-agreement protocol suggested by Rivest and Sherman in 1984, and in digital signatures as suggested by Rabi and Sherman in 1993 [14].

The rest of this paper is organized in four sections. Section 2 precisely defines the concept of an AOWF.

* Corresponding author. Email: sherman@cs.umbc.edu.
[1] Email: mrabi@cs.umbc.edu.
[2] Member, Institute for Advanced Computer Studies, University of Maryland College Park.

Section 3 proves that AOWFs exist if and only if P $\neq$ NP. Section 4 presents two possible applications of a strong form of AOWFs – the aforementioned key-agreement and digital-signature protocols. Finally, Section 5 summarizes our conclusions and lists several open problems.

2. AOWFs: Definitions

An *associative one-way function* (AOWF) is any binary function that is associative and one-way. In this section, we precisely define these structured functions. Using a worst-case model, we say that a one-way function is any function that can be computed in polynomial time but that cannot be inverted in polynomial time.

Within our definitions, we shall deal exclusively with binary functions on the infinite message space $\mathcal{S} = \{0,1\}^*$ of all finite binary strings. Let $\circ : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ be any such function. We will denote function application using infix or prefix notation (*e.g.* $x \circ y = \circ(x,y)$). Throughout, unless otherwise stated, all functions are total (e.g. domain$(\circ) = \mathcal{S} \times \mathcal{S}$). Although the definitions are straightforward, there are some important technicalities (e.g. honesty requirement) and subtleties (e.g. dealing with inverses and partial functions). For any strings $x, y \in \mathcal{S}$, let $|x|$ denote the length (i.e. number of characters) of x, and let $x||y$ denote the concatenation of x and y.

To ensure that the difficulty of inverting an AOWF not be caused simply by its input being much longer than its output, we require every AOWF to be honest in the following standard sense.

Definition 1. Any binary *function* $\circ : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ is *honest* if and only if there exists a polynomial p such that, for every $z \in$ image$(\circ)$, there exist $x, y \in \mathcal{S}$ such that $x \circ y = z$ and $|x| + |y| \leqslant p(|z|)$.

Because we do not require that AOWFs be injective (in fact, we prove they cannot be), we must explain what it means to invert a non-injective function. By inverting $\circ$ we mean: given any $z \in$ image$(\circ)$, find any $x, y \in \mathcal{S}$ such that $x \circ y = z$.

Definition 2. Any binary *function* $\circ : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ is *one-way* if and only if $\circ$ is honest; $\circ$ is computable in polynomial time; and inverting $\circ$ is not computable in polynomial time.

By associativity, we shall always mean associativity of function application. Since we prove the existence of partial AOWFs, we need to extend the usual notion of associativity to partial functions.

Definition 3. Let $\circ : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ be any partial binary function. We say $\circ$ is *associative* if and only if $x \circ (y \circ z) = (x \circ y) \circ z$. If $\circ$ is total, we require this equation to hold for all $x, y, z \in \mathcal{S}$. If $\circ$ is partial, we require this equation to hold for all $x, y, z \in \mathcal{S}$ such that each of (x,y), (y,z), $(x, y \circ z)$, and $(x \circ y, z)$ is an element of domain$(\circ)$.

Combining Definitions 1–3 yields our definition of an AOWF.

Definition 4. Any binary function $\circ : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ is an *AOWF* if and only if $\circ$ is both associative and one-way.

It is easy to verify that no AOWF is injective: given any $y, z \in \mathcal{S}$, if $\circ : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$ is injective, then $z \circ (y \circ z) = (z \circ y) \circ z$ implies that (y,z) and (z,y) are preimages of z.

As an example, we note that integer multiplication over the large odd integers is a plausible AOWF: this operation is associative and easy to compute, and its inverse problem – integer splitting – is believed to be hard.

3. Existence proof

Strengthening a theorem of Selman [17, Proposition 1], we constructively prove that AOWFs exist if and only if P $\neq$ NP. Under the hypothesis P $\neq$ NP, our main theorem proves the existence of a partial AOWF. Our construction is based on the computation tree of any polynomial-time nondeterministic Turing machine that accepts any language in NP – P. Given any NP-complete problem (e.g. three coloring), and assuming P $\neq$ NP, our construction yields a concrete AOWF. We also show how certain AOWFs can be lifted to total functions.

Let A ∈
any NP-m
man const
the functic
$\mathcal{S} = \{0, 1$
is any acce
erwise, *co*
way becau
tree upwar
In particul
tend Selm
comp func
tive functi
sociativity
ship.

Through
Turing mac
so that the i
putations a
unique.

Theorem 5
if P $\neq$ NP.

Proof. (N
AOWF is a
Selman's T

($\Leftarrow$) Ass
guage A ∈
accepts A,
tions of all
partial AOV
$C_M \to C_M$ >

First, for
be true if an
some config
tree of M o
the closest
any $x \in C_M$

acomp$_M(x)$

where $(y_0,$
scribed in th
$y_0 = y_1$. The

Now, defi
to be any in

Let $A \in \mathrm{NP} - \mathrm{P}$ be any language, and let M be any NP-machine that accepts A. In his proof, Selman constructs a one-way function as the inverse of the function $comp_M : \mathcal{S} \to \mathcal{S}$, defined for any $x \in \mathcal{S} = \{0,1\}^*$ as follows. If $x \in A$, then $comp_M(x)$ is any accepting configuration of M on input x; otherwise, $comp_M(x) = \perp$. Intuitively, $comp_M^{-1}$ is one-way because it is easy to traverse M's computation tree upwards but hard to traverse this tree downward. In particular, it is hard to decide if $x \in A$. To extend Selman's Theorem to AOWFs, we modify the *comp* function so that its inverse is a binary associative function. Our modification is based on the associativity of the closest common ancestor relationship.

Throughout, we assume that each nondeterministic Turing machine writes its guesses on a separate tape, so that the instantaneous descriptions of any two computations along different nondeterministic paths are unique.

Theorem 5. *There exists a partial AOWF if and only if* $\mathrm{P} \neq \mathrm{NP}$.

Proof. (Necessity and sufficiency.) ($\Rightarrow$) Since every AOWF is a one-way function, the proof follows from Selman's Theorem.

($\Leftarrow$) Assume $\mathrm{P} \neq \mathrm{NP}$. Then there exists some language $A \in \mathrm{NP} - \mathrm{P}$. Let M be any NP-machine that accepts A, and let $\mathcal{C}_M$ denote the set of all configurations of all computations of M. We will construct a partial AOWF as any inverse of the function $acomp_M : \mathcal{C}_M \to \mathcal{C}_M \times \mathcal{C}_M$, which we will now define.

First, for any $x \in \mathcal{C}_M$, define the predicate $\Phi_M(x)$ to be true if and only if there exist some string $w \in A$ and some configurations $y_0, y_1 \in \mathcal{C}_M$ in the computation tree of M on input w such that $x \notin \{y_0, y_1\}$ and x is the closest common ancestor of y_0 and y_1. Then, for any $x \in \mathcal{C}_M$, define

$$acomp_M(x) = \begin{cases} (y_0, y_1) & \text{if } \Phi_M(x) \text{ is true,} \\ \perp & \text{otherwise,} \end{cases} \qquad (1)$$

where (y_0, y_1) is any pair of configurations as described in the definition of $\Phi_M(x)$. It is possible that $y_0 = y_1$. The symbol $\perp$ means undefined.

Now, define the partial function $f : \mathcal{C}_M \times \mathcal{C}_M \to \mathcal{C}_M$ to be any inverse of $acomp_M$. We will prove that f is honest; f is associative; f is computable in polynomial time; and f cannot be inverted in polynomial time.

Claim 1. *f is honest.*

We must show that there exists some polynomial p such that, for all $x \in \mathcal{C}_M$, $|acomp_M(x)| \leqslant p(|x|)$. This inequality holds for p being twice the running time of M. It is true that M runs in polynomial time, and no configuration can be larger than the time needed to compute it. Thus, f is honest.

Claim 2. *f is associative.*

Let x, y, z be any configurations in $\mathcal{C}_M$ such that each of (x,y), (y,z), $(x, f(y,z))$, and $(f(x,y), z)$ is an element of domain(f). By the definition of associativity, we must prove $f(x, f(y,z)) = f(f(x,y),z)$; that is, we must prove $f(w_0, z) = f(x, w_1)$, where $w_0 = f(x,y)$ and $w_1 = f(y,z)$. By the definition of f, there exists some $w \in A$ such that w_0 is the closest common ancestor of x and y, and w_1 is the closest common ancestor of y and z, along some computation paths in the computation tree of M on input w. It follows that $f(w_0, z) = f(x, w_1)$ since this configuration is the closest common ancestor of w_0 and w_1.

Claim 3. *f is computable in polynomial time.*

Let (y_0, y_1) be any configurations in domain(f). Thus, there exists $w \in A$ such that y_0 and y_1 are configurations along some computation paths in the computation tree of M on input w. Since M runs in polynomial time, these paths are at most polynomially long. By traversing these paths upwards, $f(y_0, y_1)$ can be computed in polynomial time as the closest common ancestor of y_0 and y_1. Hence, f is computable in polynomial time, even though recognizing domain(f) might take longer.

Claim 4. *f^{-1} is not computable in polynomial time.*

To compute f^{-1} is to compute $acomp_M$. Were $acomp_M$ polynomial-time computable, we could decide A in polynomial time as follows. Given any input string b, let x_b be any child of the initial configuration of the computation of M on input b. Then $b \in A$ if and only if $acomp_M(x_b) \neq \perp$. Since $A \notin$ P, $acomp_M$ is not computable in polynomial time. □

The AOWF f constructed in the proof of Theorem 5 is commutative, because the closest common ancestor relation is commutative.

Although recognizing domain(f) is as hard as recognizing A, given any AOWF g with domain$(g) \in$ P, it is possible to extend g to a total AOWF $\widehat{g}$ in a straightforward fashion. Specifically, let $c \in \mathcal{C}_M$ be any string such that $(c,c) \notin$ domain(g). Then define $\widehat{g}(x,y) = g(x,y)$ whenever $(x,y) \in$ domain(g), and $\widehat{g}(x,y) = c$ otherwise.

Our theorem relates to previous work in worst-case cryptocomplexity as follows. In 1979, Brassard [1] proved that the existence of a one-way function would imply P $\subsetneq$ NP $\cap$ coNP and thus P $\neq$ NP. Brassard required that his one-way function be bijective and that its image be in coNP. In 1984, Grollmann and Selman ([6] and [17, Proposition 3]) tightened this result by proving that an *injective* one-way function exists if and only if P $\neq$ UP, where UP (unique P) is the class of languages that can be accepted in nondeterministic polynomial time with unique accepting paths.[3] Dropping the injectivity requirement, Selman [17] also proved that one-way functions exist if and only if P $\neq$ NP. Thus, the existence of an injective one-way function depends on a stronger (and apparently much stronger) complexity assumption than does the existence of a one-way function. Since AOWFs cannot be injective, we began with Selman's second theorem.

4. Applications

As evidence of the possible utility of AOWFs, we present two novel protocols that apply AOWFs to achieve secret-key agreement and digital signatures. Each of the protocols requires a *strong AOWF* – one that is difficult to invert, even if either one of its inputs is given.

By inverting $\circ$ given its second argument, we mean inverting the restricted function $\circ_y = \circ(\cdot, y)$; that is, given any $y \in \mathcal{S}$ and any $z \in$ image$(\circ_y)$, find any $x \in \mathcal{S}$ such that $x \circ y = z$. Inverting $\circ$ given its first argument is similarly defined.

The function f constructed in Theorem 5 is not a strong AOWF because, given any $x, y_1 \in \mathcal{C}_M$ such that x is in the image of f restricted to second argument y_1, it follows that $f(y_1, y_1) = x$. Similarly, multiplication of large odd integers is not strongly one-way. We conjecture, however, that strong AOWFs exist, even in average-case models of complexity.

4.1. *Key agreement*

Secret-key agreement is a fundamental problem in communications security that is of significant interest to practitioners and theorists. In its simplest form, this problem is for two parties (say, Alice and Bob), to agree on a secret key k by sending messages back and forth over an insecure channel which is wiretapped by a passive eavesdropper (say, Eve). The main security requirement is that Eve must not have more than an ε-advantage in guessing k listening to the communications over guessing k without listening to the communications, where ε is some small positive real number (say, $\varepsilon = 0.01$). We present a new solution to this problem based on the novel paradigm of strong associative one-way functions.

Throughout, we focus on unauthenticated-key agreement. Additional mechanisms are needed if Bob is to convince himself that he has been communicating with Alice; often these extra mechanisms are easy to add.

Protocol KAP given below (due to Rivest and Sherman) shows how Alice and Bob can agree on a secret key from the set $\mathcal{M} = \{0,1\}^n$, where n is some positive integer.

1. Alice generates two random numbers x and y. Alice keeps x secret and sends y and $x \circ y$ to Bob.
2. Bob generates a random number z. Bob keeps z secret and sends $y \circ z$ back to Alice.
3. Alice computes $k_A = x \circ (y \circ z)$ and Bob computes $k_B = (x \circ y) \circ z$. Alice and Bob agree on $k_A = k_B$ as their secret key.

The protocol exploits a strong AOWF $\circ : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$, known to Alice, Bob, and the enemy Eve. During the protocol, random numbers x, y, z are selected from the set $\mathcal{M}$. In the last step of the protocol, Alice computes her key $k_A = x \circ (y \circ z)$ and Bob

[3] According to Selman [17], the essence of this result was independently discovered by Berman in 1977.

computes his key $k_B = (x \circ y) \circ z$; the associativity of $\circ$ ensures that $k_A = k_B$.

At the end of the protocol Eve knows y, $x \circ y$, and $y \circ z$. For the protocol to be secure, from this information, Eve must not be able to guess the agreed-upon key $x \circ y \circ z$ with advantage. Our assumptions are that $\circ$ is a strong AOWF on $\mathcal{M} = \{0, 1\}^n$ and that x, y, z are chosen independently with uniform distribution from $\mathcal{M}$.

If Eve could compute x or z, then Eve could compute the key as $x \circ (y \circ z)$ or $(x \circ y) \circ z$. This direct attack is impossible because it would contradict the assumption that $\circ$ is a strong AOWF. Thus, the only way in which KAP could possibly fail is if Eve could compute $x \circ y \circ z$ without computing x or z.

One possible attack would be to compute $x \circ y \circ z$ by applying $\circ$ to some sequence of terms drawn from the given values y, $x \circ y$, and $y \circ z$. For example, if $y^r = y$ for some $r - 1$ applications of $\circ$, then Eve could compute $x \circ y \circ z = (x \circ y) \circ y^{r-2} \circ (y \circ z)$. Thus, for KAP to be secure, it must not be true that $y^r = y$ for some polynomially-bounded r.

The key-agreement problem was introduced in 1976 by Diffie and Hellman [4,5], who referred to it as the public-key distribution problem. Without any proof of security, Diffie and Hellman also suggested a solution based on the difficulty of computing discrete logarithms over GF(p). Despite some similarities with Protocol KAP, the Diffie–Hellman scheme is not based on an AOWF [14]. Since 1976, several other approaches to key agreement have been proposed, many of which are surveyed by Rueppel and van Oorschot [16]. Also, in 1979, Ingemarsson [8] studied algebraic properties of sets of functions that make up public-key distribution systems.

Although several approaches to proving the security of key-agreement protocols have been tried (e.g. see Maurer [11,12], Desmedt and Burmester [3], and van Oorschot [13]), no one has proven the security of the Diffie–Hellman scheme, even assuming that computing discrete logarithms is infeasible in polynomial time. We, too, leave the security of our protocol to the scrutiny of the reader.

Whenever the strong AOWF $\circ$ is commutative, Protocol KAP can be generalized to enable any N parties to agree on a secret key [14]. The method is similar to that suggested by Ingemarsson, Tang, and Wong [9] for the Diffie–Hellman scheme.

Any public-key cryptosystem can be used to effect secret-key agreement, but practitioners have sought solutions that do not require or make available the additional mechanisms and capabilities of public-key cryptosystems. Our approach to key agreement offers a new cryptographic building block – strong AOWFs.

4.2. Digital signatures

Strong AOWFs can also be used to sign messages. To this end, let $\circ : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be any strong AOWF, and assume that there is an authenticated public directory. Initially, each user U generates two numbers $x_U, y_U \in \mathcal{M}$ at random, keeps x_U secret, and places the pair $(y_U, x_U \circ y_U)$ into the public directory. To sign any message $m \in \mathcal{M}$, the user computes the signature $\sigma_U(m) = m \circ x_U$. To verify any purported message-signature pair (m, σ) from U, the recipient retrieves y_U and $x_U \circ y_U$ from the public directory and computes $\sigma \circ y_U$ and $m \circ (x_U \circ y_U)$. The recipient accepts σ as a valid signature of m by U if and only if $\sigma \circ y_U = m \circ (x_U \circ y_U)$.

As with many other signature schemes, this scheme is vulnerable to what Rivest [15] calls existential forgery: given any valid message-signature pair $(m, \sigma_U(m))$, it is possible to forge signatures of new messages of the form $m' = z \circ m$, for any $z \in \mathcal{M}$. Specifically, forge $\sigma_U(m') = m' \circ x_U = (z \circ m) \circ x_U$ by computing $\sigma_U(m') = z \circ \sigma_U(m) = z \circ (m \circ x_U)$. To overcome this difficulty, one should use a public cryptographically-secure hash function, as suggested by Davies and Price [2] and as is typically done in many signature systems. When using a hash function $h : \mathcal{M} \to \mathcal{M}$, the signer would compute the signature $h(m) \circ x_U$ and assume that Eve cannot find any $z \in \mathcal{M}$ and any intelligible message $m' \in \mathcal{M}$ such that $h(m') = z \circ h(m)$.

5. Conclusion and open problems

We have introduced the concept of associative one-way functions (AOWFs) as an intriguing and useful new cryptographic paradigm. We proved that partial AOWFs exist if and only P $\neq$ NP, and we presented protocols for applying strong AOWFs to reach unauthenticated secret-key agreement and to sign documents.

AOWFs illustrate a beneficial synergism that can ensue when a cryptographic object is endowed with a combination of algebraic and security properties. To explore such combinations, AOWFs are a natural place to start because they combine two of the most fundamental properties from algebra and cryptographic security: associativity and one-wayness.

We conclude with five open problems.

(1) Do strong AOWFs exist?
(2) Exhibit a plausible strong AOWF.
(3) Prove (or disprove) that protocol KAP is secure.
(4) What can be said about the distribution of the agreed-upon key in protocol KAP?[4] In particular, is it possible to modify Protocol KAP to ensure that neither Alice nor Bob alone can bias the distribution of the agreed-upon key?
(5) What other applications do AOWFs have?

All of these questions would be particularly interesting to answer in average-case models of complexity, such as those studied by Impagliazzo and Rudich [7].

Although the security of Protocol KAP remains open, so does that of the Diffie–Hellman protocol. Nevertheless, Protocol KAP and our digital signature method are evidence that AOWFs – and more generally, functions that combine fundamental algebraic and security properties – offer elegant solutions to a variety of practical cryptographic problems.

Acknowledgments

We thank Thomas Cain, Richard Chang, Ronald Rivest, an Ali Selçuk for helpful editorial comments.

References

[1] G. Brassard, A note on the complexity of cryptography, IEEE Trans. Inform. Theory 25 (2) (1979) 232–233.
[2] D.W. Davies, W.L. Price, The application of digital signatures based on public-key cryptosystems, in: Proc. 5th Internat. Computer Communications Conf., 1980, pp. 525–530.
[3] Y. Desmedt, M. Burmester, Towards practical 'proven secure' authenticated key distribution, in: Proc. 1st ACM Conf. on Computer and Communications Security, ACM Press, New York, 1993, pp. 228–231.
[4] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory 22 (6) (1976) 644–654.
[5] W. Diffie, M.E. Hellman, Privacy and authentication: An introduction to cryptography, Proc. IEEE 67 (3) (1979) 397–427.
[6] J. Grollman, A. Selman, Complexity measures for public-key cryptosystems, SIAM J. Comput. 17 (2) (1988) 309–335.
[7] R. Impagliazzo, S. Rudich, Limits on the provable consequences of one-way permutations, in: Proc. 21st Ann. Symp. on Theory of Computation, ACM Press, New York, 1989, pp. 44–61 (to appear in: J. Cryptology).
[8] I. Ingemarsson, The algebraic structure of public-key distributions systems, Tech. Rept. 1979-02-08, LiTH-ISY-I-2070, Dept. of Electrical Engineering, Linköping University, 1979, 12 pp.
[9] I. Ingemarsson, D.T. Tang, C.K. Wong, A conference key distribution system, IEEE Trans. Inform. Theory 28 (5) (1982) 714–719.
[10] B.S. Kaliski Jr, R.L. Rivest, A.T. Sherman, Is the Data Encryption Standard a group? (Results of cycling experiments on DES), J. Cryptology 1 (1) (1988) 3–36.
[11] U.M. Maurer, Towards the equivalence of breaking the Diffie–Hellman protocol and computing discrete logarithms, in: Y.G. Desmedt (Ed.), Advances in Cryptology: Proc. Crypto 94, Lecture Notes in Computer Science, vol. 839, Springer, Berlin, 1994, pp. 271–281.
[12] U.M. Maurer, S. Wolf, Diffie–Hellman oracles, in: N. Koblitz (Ed.), Advances in Cryptology: Proc. Crypto 96, Lecture Notes in Computer Science, vol. 1109, Springer, Berlin, 1994, pp. 268–282.
[13] P.C. van Oorschot, Extending cryptographic logics of belief to key agreement protocols (extended abstract), in: Proc. 1st ACM Conf. on Computer and Communications Security, ACM Press, New York, 1993, pp. 232–243.
[14] M. Rabi, A.T. Sherman, Associative one-way functions: A new paradigm for secret-key agreement and digital signatures, Tech. Rept. CS-TR-3183/UMIACS-TR-93-124, University of Maryland College Park, 1993, and Tech. Rept. TR CS-93-18, Computer Science Dept., University of Maryland Baltimore County, 1993, 13 pp. (http://www.cs.umbc.edu/~sherman).
[15] R.L. Rivest, Cryptography, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science. Volume A: Algorithms and Complexity, Elsevier, Amsterdam, 1990, Chapter 13, pp. 717–755.
[16] R.A. Rueppel, P.C. van Oorschot, Modern key agreement techniques, Computer Communications 17 (7) (1994) 458–465.
[17] A.L. Selman, A survey of one-way functions in complexity theory, Math. Systems Theory 25 (3) (1992) 203–221.
[18] A.T. Sherman, Cryptology and VLSI (a two-part dissertation), Tech. Rept. MIT/LCS/TR-381, MIT Laboratory for Computer Science, 1986.
[19] C.P. Waldvogel, J.L. Massey, The probability distribution of the Diffie–Hellman key, in: J. Seberry, Y. Zheng (Eds.), Proc. AUSCRYPT 92, Lecture Notes in Computer Science, vol. 718, Springer, Berlin, 1993, pp. 492–504.

[4] For some related work on the Diffie–Hellman scheme, see Waldvogel and Massey [19].

Editor

Th
results
concise
equival
Th
found
most
least
Editors
on expe
The
of keyw

Algorit
combin
comput
cryptog
distribu
formal
informa
systems
evaluati
program
specific
– softw
specific

Instruc

Con
the Boa
area pr
Since
provide
The
etc.) sh
pages,
approxi
decision
An
supplied
list of
put in
should
article,
(or title
page m
preferred
Sinc
no need
used, ca
paginate
earlier.
size or,
Once
correspo
Science
Netherla
proofrea
to the
extremel
should
text.
Upon
copyrigh
will ensu
Then
each art
Publisher
Notice w

Instructi

The
publicatio
a disket