New Public-key Cryptosystem Using Braid Groups

Ki Hyoung Ko[1], Sang Jin Lee[1], Jung Hee Cheon[2],
Jae Woo Han[3], Ju-sung Kang[3], and Choonsik Park[3]

[1] Department of Mathematics, Korea Advanced Institute of Science and Technology, Taejon, 305-701, Korea
{knot,sjlee}@knot.kaist.ac.kr
[2] Department of Mathematics, Brown university, Providence, RI 02912, USA
and Securepia, Korea
jhcheon@math.brown.edu
[3] Section 8100, Electronics and Telecommunications Research Institute, Taejon, 305-600, Korea
{jwhan,jskang,csp}@etri.re.kr

Abstract. The braid groups are infinite non-commutative groups naturally arising from geometric braids. The aim of this article is twofold. One is to show that the braid groups can serve as a good source to enrich cryptography. The feature that makes the braid groups useful to cryptography includes the followings: (i) The word problem is solved via a fast algorithm which computes the canonical form which can be efficiently manipulated by computers. (ii) The group operations can be performed efficiently. (iii) The braid groups have many mathematically hard problems that can be utilized to design cryptographic primitives.
The other is to propose and implement a new key agreement scheme and public key cryptosystem based on these primitives in the braid groups. The efficiency of our systems is demonstrated by their speed and information rate. The security of our systems is based on topological, combinatorial and group-theoretical problems that are intractible according to our current mathematical knowledge. The foundation of our systems is quite different from widely used cryptosystems based on number theory, but there are some similarities in design.

Key words: public key cryptosystem, braid group, conjugacy problem, key exchange, hard problem, non-commutative group, one-way function, public key infrastructure

1 Introduction

1.1 Background and previous results

Since Diffie and Hellman first presented a public-key cryptosystem(PKC) in [11] using a trapdoor one-way function, many PKC's have been proposed and broken.

Most of successful PKC's require large prime numbers. The difficulty of factorization of integers with large prime factors forms the ground of RSA [29] and its variants such as Rabin-Williams [28, 36], LUC's scheme [32] or elliptic curve versions of RSA like KMOV [20]. Also the difficulty of the discrete logarithm problem forms the ground of Diffie-Hellman type schemes like ElGamal [12], elliptic curve cryptosystem, DSS, McCurley [23].

There have been several efforts to develop alternative PKC's that are not based on number theory. The first attempt was to use NP-hard problems in combinatorics like Merkle-Hellman Knapsack [24] and its modifications. Though many cryptographers have been pessimistic about combinatorial cryptography after the breakdown of the Knapsack-type PKC's by Shamir [30], Brickell [9], Lagarias [22], Odlyzko [26], Vaudenay [35] and others, and after the appearance of Brassard theorem [8], there may still be some hopes as Koblitz has noted in [21]. The other systems that are worth to mention are the quantum cryptography proposed by Bennet and Brassard [4] and the lattice cryptography proposed by Goldreich, Goldwasser and Halevi [18].

Another approach is to use hard problems in combinatorial group theory such as the word problem [1, 37, 17] or using the Lyndon words [31]. Recently Anshel-Anshel-Goldfeld proposed in [2] a key agreement system and a PKC using groups where the word problem is easy but the conjugacy problem is intractible. And they noted that the usage of braid groups is particularly promising. Our proposed systems is based on the braid groups but is independent from their algebraic key establishment protocol on monoids in [2].

Most of cryptosystems derived from combinatorial group theory are mainly theoretical or have certain limitations in wide and general practice. This is perhaps due to the rack of efficient description of group elements and operations or due to the difficulty of implementing cryptosystems themselves.

1.2 The features of braid groups

The n-braid group B_n is an infinite noncommutative group of n-braids defined for each positive integer n. There is a natural projection from B_n to the group Σ_n of all $n!$ n-permutations and so B_n can be thought as a resolution of Σ_n. In this article, we first show that the braid groups have the following nice properties, unlike the usual combinatorial groups, so that one can build cryptosystems satisfying both security and efficiency requirements.

1. There is a natural way to describe group elements as data which can be handled by computers: Theorem 1 shows that there is a canonical form for a braid, which can be described as an ordered tuple $(u, \pi_1, \pi_2, \ldots, \pi_p)$, where u is an integer and π_i's are n-permutations. The canonical form can remove the difficulties in using words in the description of the group elements.
2. There are fast algorithms to perform the group operations: The product of two words U and V is just the concatenation UV and therefore the group operation for the purpose of cryptography really means hiding the factors U and V. This can be achieved by converting UV into its canonical form. For

a group whose element has no canonical form, this can be achieved only by rewriting via defining relations and a retrieval must be done by a solution to the word problem. Let U and V be n-braids whose canonical forms are represented by p and q permutations respectively. Theorem 2 shows that the canonical form of the product UV can be computed in time $\mathcal{O}(pqn \log n)$ and the canonical form of the inverse of U can be computed in time $\mathcal{O}(pn)$

3. There are many hard problems based on topological or group-theoretical open problems and one can sometimes design (trap-door) one-way functions based on these problems that can be described basically by group operations.
4. As n grows in the braid groups B_n, the computation of group operations become harder in $\mathcal{O}(n \log n)$. On the other hand, a naive computation of the inverses of one-way functions are seem to be at least $\mathcal{O}(n!)$. Consequently, n plays a reliable role of a security parameter.

1.3 Our results

After exploring cryptographic aspects of the braid groups in §2, we propose a trapdoor one-way function that is based on one of the hard problems in §2.3 and construct a key exchange scheme and a public-key cryptosystem in §3. A theoretic operating characteristics and implementation of our PKC will be given in §4 and so they are readily available in practice. Our PKC has the following features.

1. Our key exchange scheme is based on a variation of the conjugacy problem similar to the Diffie-Hellman problem and our PKC is constructed from this key exchange scheme. Therefore our PKC and behaves somewhat similarly to ElGamal PKC.
2. Our PKC is non-deterministic: The ciphertext depends on both of the plaintext and the braid chosen randomly at each session.
3. The message expansion is at most 4-1.
4. There are two parameters p, n in our PKC so that the message length becomes $pn \log n$. The encryption and decryption are $\mathcal{O}(p^2 n \log n)$ operations. The security level against brute force attacks is $\mathcal{O}((n!)^p) = \mathcal{O}(\exp(pn \log n))$. Thus the parameter n rapidly increases the security level without sacrificing the speed.

Our cryptosystems are efficient enough, comparing to other widely used cryptosystems. The security of our scheme is discussed and a possible attack based on a mathematical knowledge is introduced in §5. As a further study, possible improvements of our cryptosystems and possible replacements of the braid groups are discussed in §6.

2 A cryptographic aspect of the braid groups

The braid group was first introduced by Artin in [3]. Because these groups play important roles in low dimensional topology, combinatorial group theory and

(a) the 3-braid $\sigma_2^2\sigma_1^{-1}\sigma_2$ (b) the generator σ_i

Fig. 1. An example of braid and the generator

representation theory, considerable research has been done on these groups. In this section, we will briefly introduce the notion of braids and give evidence that the braid groups can also play important roles in cryptography. The general reference for braid theory is the Birman's book [5] and for the word problem and conjugacy problem, see [6, 13, 14, 16].

This section is composed as follows: §2.1 is the definition of the braid groups. In §2.2 we first summarize the known results on the word problem (or the canonical form problem). Theorem 1 is important since it enables one to encode a braid into a data format that can be handled easily by computers. The remains are supplementary to this theorem.

In §2.3 we list hard problems that are potential sources to develop primitives in cryptography.

2.1 Definition of the n-braid group

The n-braid group B_n is defined by the following group presentation.

$$B_n = \left\langle \sigma_1, \ldots, \sigma_{n-1} \;\middle|\; \begin{array}{ll} \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j & \text{if } |i-j| = 1 \\ \sigma_i\sigma_j = \sigma_j\sigma_i & \text{if } |i-j| \geq 2 \end{array} \right\rangle \qquad (*)$$

The integer n is called the *braid index* and each element of B_n is called an n-*braid*. Braids have the following geometric interpretation: an n-braid is a set of disjoint n strands all of which are attached to two horizontal bars at the top and at the bottom such that each strand always heads downward as one walks along the strand from the top to the bottom. The braid index is the number of strings. See Figure 1(a) for an example. Two braids are *equivalent* if one can be deformed to the other continuously in the set of braids. In this geometric interpretation, σ_i is the elementary braid as in Figure 1(b).

The multiplication ab of two braids a and b is the braid obtained by positioning a on the top of b. The identity e is the braid consisting of n straight vertical strands and the inverse of a is the reflection of a with respect to a horizontal line. So σ_i^{-1} can be obtained from σ_i by switching the over-strand and under-strand. See [5] for details.

Note that if we add the relation $\sigma_i^2 = 1$, $i = 1, \ldots, n-1$, to the presentation $(*)$, it becomes the group presentation of the n-permutation group Σ_n, where σ_i corresponds to the transition $(i, i+1)$. So there is a natural surjective homomorphism $\rho\colon B_n \to \Sigma_n$. Let's denote a permutation $\pi \in \Sigma_n$, $\pi(i) = b_i$, by $\pi = b_1 b_2 \cdots b_n$. In terms of geometric braids, the homomorphism $\rho\colon B_n \to \Sigma_n$ can be described as follows: given a braid a, let the strand starting from the i-th upper position ends at the b_i-th lower position. Then $\rho(a)$ is the permutation $b_1 b_2 \cdots b_n$.

2.2 Describing braids using permutations

The easiest way to describe a braid is to write it as a word on σ_i's. But there is no unique way to do this. For example, all the words in the following formula represent the same braid Δ_4 in Figure 2(a), where the defining relations, $\sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma$, $\sigma_2\sigma_3\sigma_2 = \sigma_3\sigma_2\sigma_3$ and $\sigma_1\sigma_3 = \sigma_3\sigma_1$, are applied to the underlined subwords.

$$\begin{aligned}\Delta_4 &= \sigma_1\sigma_2\sigma_3\underline{\sigma_1\sigma_2\sigma_1} = \sigma_1\underline{\sigma_2\sigma_3\sigma_2}\sigma_1\sigma_2 = \underline{\sigma_1\sigma_3}\sigma_2\underline{\sigma_3\sigma_1}\sigma_2 \\ &= \sigma_3\underline{\sigma_1\sigma_2\sigma_1}\sigma_3\sigma_2 = \sigma_3\sigma_2\sigma_1\underline{\sigma_2\sigma_3\sigma_2} = \sigma_3\sigma_2\sigma_1\sigma_3\sigma_2\sigma_3\end{aligned}$$

In 1947, Artin proved that a braid can be described uniquely as an automorphism of the free group of rank n [3]. In late sixties, Garsides solved the word problem after exploring the properties of the semigroup of positive words in [16] and his idea was improved by Thurston [14], Elrifai-Morton [13] and Birman-Ko-Lee [6]. They showed that there is a fast algorithm to compute the canonical form, which is unique for their results briefly.

Before stating the theorem, we introduce the notions of the *permutation braid* and the *fundamental braid*. To each permutation $\pi = b_1 b_2 \cdots b_n$, we associate an n-braid A that is obtained by connecting the upper i-th point to the lower b_i-th point by a straight line and then making each crossing positive, i.e. the line between i and b_i is under the line between j and b_j if $i < j$. For example if $\pi = 4213$, then $A = \sigma_1\sigma_2\sigma_1\sigma_3$ as in Figure 2(b).

The braids made as above is called a *permutation braid* or a *canonical factor* and $\tilde{\Sigma}_n$ denotes the set of all permutation braids. The correspondence from a permutation π to a canonical factor A is a right inverse of $\rho\colon B_n \to \Sigma_n$ as a set function. So the cardinality of $\tilde{\Sigma}_n$ is $n!$. The permutation braid can be characterized by the property that every crossing is positive and any pair of strands crosses at most once [13, 14].

The permutation braid corresponding to the permutation $\Omega_n = n(n-1)\cdots(2)1$ is called the *fundamental braid* and denoted by Δ_n. If there is no confusion on the braid index, we drop the subscript to write just Δ. See Figure 2(a) for an example.

The following theorem gives a method to describe a braid. It is Theorem 2.9 of [13], where they proved the theorem for the positive words, and then discuss the general words in the next section. The notion of left-weightedness will be explained right after.

(a) the fundamental braid Δ_4 (b) the braid $A \in \tilde{\Sigma}_n$ corresponding to $\pi = 3124$

Fig. 2. The permutation braid and the fundamental braid

Theorem 1. *For any $W \in B_n$, there is a unique representation, called the* left-canonical form*, as*

$$W = \Delta^u A_1 A_2 \cdots A_p, \qquad u \in \mathbb{Z},\ \ A_i \in \tilde{\Sigma}_n \setminus \{e, \Delta\},$$

where $A_i A_{i+1}$ is left-weighted for $1 \leq i \leq p-1$.

So we can describe a braid $W = \Delta^u A_1 A_2 \cdots A_p$ by a tuple $(u, \pi_1, \pi_2, \ldots, \pi_p)$, where the canonical factor A_i corresponds to the permutation π_i. Here p is called the *canonical length*, denoted by $\mathrm{len}(W)$, of W. We will use this description when implementing cryptosystems. Now we explain briefly the idea of Garside, Thurston, Elrifai-Morton and Birman-Ko-Lee, following the paper of Elrifai-Morton [13].

1. Note that the relations in the group presentation $(*)$ relate two positive words with same word length. Let B_n^+ be the semigroup defined by the generators and relations in the presentation. Garside proved that the natural homomorphism $: B_n^+ \to B_n$ is injective [16]. Thus two positive words P and Q are equivalent in B_n if and only if P and Q are equivalent in B_n^+.
2. For a positive word P, the *starting set* $S(P)$ and the *finishing set* $F(P)$ is defined by

$$S(P) = \{i \mid P = \sigma_i P' \text{ for some } P' \in B_n^+\},$$
$$F(P) = \{i \mid P = Q'\sigma_i \text{ for some } Q' \in B_n^+\}.$$

   For a canonical factor A corresponding to a permutation $\pi \in \Sigma_n$, $S(A) = \{i \mid \pi(i) > \pi(i+1)\}$ [14] and similarly $F(A) = \{i \mid \pi^{-1}(i) > \pi^{-1}(i+1)\}$. We note that $S(A)$ is just the descent set [33] defined in the combinatorics. So for Δ_4 in Figure 2(a), $S(\Delta_4) = F(\Delta_4) = \{1, 2, 3\}$ and for $A = \sigma_1\sigma_2\sigma_1\sigma_3$ in Figure 2(b), $S(A) = \{1, 2\}$ and $F(A) = \{1, 3\}$.
3. The fundamental braid Δ has the following two properties.
   (a) For each $1 \leq i \leq n-1$, $\Delta = \sigma_i A_i = B_i \sigma_i$ for some permutation braids A_i and B_i.
   (b) For any $1 \leq i \leq n-1$, $\sigma_i \Delta = \Delta \sigma_{n-i}$.

For an arbitrary word W on σ_i's, we can replace each occurrence of σ_i^{-1} by the formula $\sigma_i^{-1} = \Delta^{-1}B_i$ from the first property and collect Δ^{-1}'s to the left by using the second property to get the expression $W = \Delta^u P$, $P \in B_n^+$.

4. For any positive word P, there is a unique decomposition, which is called the *left-weighted decomposition* as follows:

$$P = A_1 P_1, \qquad A_1 \in \tilde{\Sigma}_n, \;\; P_1 \in B_n^+, \;\; F(A_1) \supset S(P_1).$$

By iterating the left-weighted decomposition $P = A_1P_1$, $P_1 = A_2P_2$, ..., and then collecting Δ's to the left, we have the *left-canonical form*

$$P = \Delta^u A_1 A_2 \cdots A_p, \qquad u \in \mathbb{Z}, \;\; A_i \in \tilde{\Sigma}_n \setminus \{e, \Delta\},$$

where A_iA_{i+1} is left-weighted. (In fact, A_iA_{i+1} is left-weighted for all $1 \leq i < p$ if and only if $A_i(A_{i+1} \cdots A_p)$ is left-weighted for all $1 \leq i < p$.) *This left canonical form is unique.*

5. By combining 3 and 4, we have the left-canonical form for arbitrary braids as in Theorem 1.

Theorem 2. *1. Let W be a word on σ_i's with word length ℓ. Then the left-canonical form of W can be computed in time $\mathcal{O}(\ell^2 n \log n)$.*
2. Let $U = \Delta^u A_1 \cdots A_p$ and $V = \Delta^v B_1 \cdots B_q$ be the left-canonical forms of n-braids. Then we can compute the left-canonical form of UV in time $\mathcal{O}(pqn \log n)$.
3. If $U = \Delta^u A_1 \cdots A_p$ is the left-canonical form of $U \in B_n$, then we can compute the left-canonical form of U^{-1} in time $\mathcal{O}(pn)$.

Proof. The proofs of 1 and 2 are in [14]. The left-canonical form of U^{-1} is given by $U^{-1} = \Delta^{-u-p} A'_p \cdots A'_1$, where A'_i is the permutation braid such that $\Delta = A_i(\Delta^{u+i} A'_i \Delta^{-u-i})$ [13]. Let π_i and Ω be the permutations corresponding to A_i and Δ. Then A'_i is the permutation braid corresponding to the permutation $\pi' = \Omega^{-u-i}(\pi_i^{-1}\Omega)\Omega^{u+i}$. Since Ω^2 is the identity, we can compute π'_i in $\mathcal{O}(n)$. Thus we can compute the whole left-canonical form of U^{-1} in time $\mathcal{O}(pn)$ as desired. □

In order to analyze the security against brute force attacks, we will need a lower bound for the number of n-braids of a given canonical length. The estimate given in the following theorem has some room to improve.

Theorem 3. *The number of n-braids of canonical length p is at least $(\lfloor \frac{n-1}{2} \rfloor !)^p$.*

Proof. Since $\lfloor \frac{n-1}{2} \rfloor = r$ for $n = 2r + 1$ and $n = 2r + 2$, and clearly there are more $(2r + 2)$-braids than $(2r + 1)$-braids of a fixed canonical length, we may assume that $n = 2r + 1$. Consider the two subsets of the $\tilde{\Sigma}_n$.

$$\mathcal{S} = \Big\{ A \in \tilde{\Sigma}_n \mid S(A) \subset \{1, 2, \ldots, r\} \text{ and } F(A) \supset \{2, 4, \ldots, 2r\} \Big\}$$
$$\mathcal{T} = \Big\{ A \in \tilde{\Sigma}_n \mid S(A) \subset \{2, 4, \ldots, 2r\} \text{ and } F(A) \supset \{1, 2, \ldots, r\} \Big\}$$

(a) The construction of element of $\mathcal{S}$ (b) The construction of element of $\mathcal{T}$

Fig. 3.

We will show that there are injective functions from $\Sigma_r \to \mathcal{S}$ and $\Sigma_r \to \mathcal{T}$. It is easy to see that the functions are not surjective and so $|\mathcal{S}| > r!$ and $|\mathcal{T}| > r!$. Since for any $A \in \mathcal{S}$ and $B \in \mathcal{T}$, AB and BA are left-weighted, there are at least $(r!)^p$ n-braids of canonical length p and so we are done.

For a permutation $\pi' \in \Sigma_r$, we construct a canonical factor A in $\mathcal{S}$ by defining the corresponding permutation $\pi \in \Sigma_{2r+1}$ by

$$\pi(i) = \begin{cases} 2\pi'(i) + 1 & \text{for } 1 \le i \le r, \\ 1 & \text{if } i = r+1, \\ 2(i-r-1) & \text{for } r+2 \le i \le 2r+1. \end{cases}$$

See Figure 3(a) for an illustration for $r = 3$. For $r+1 \le i \le 2r$, $\pi(i) < \pi(i+1)$ so that $i \notin S(A)$. Thus $S(A) \subset \{1, 2, \ldots, r\}$. And since $\pi^{-1}(2i) \ge r+1$ and $\pi^{-1}(2i+1) \le r$, $2i \in F(A)$ for any $1 \le i \le r$. So $F(A) \supset \{2, 4, \ldots, 2r\}$.

Similarly, for a permutation $\pi' \in \Sigma_r$, we construct a canonical factor A in $\mathcal{T}$ by defining the corresponding permutation $\pi \in \Sigma_{2r+1}$ by

$$\begin{aligned} \pi(2i-1) &= (r+2) - i \quad \text{for } 1 \le i \le r+1, \\ \pi(2i) &= (r+1) + \pi'(i) \quad \text{for } 1 \le i \le r. \end{aligned}$$

See Figure 3(b) for an illustration for $r = 3$. Since $\pi(2i-1) = (r+2) - i \le r+1$ and $\pi(2i) = (r+1) + \pi'(i) > r+1$, $(2i-1) \notin S(A)$ for $1 \le i \le r$. Thus $S(A) \subset \{2, 4, \ldots, 2r\}$. And since $\pi^{-1}(i) = 2r - 2i + 3 > 2r - 2i + 1 = \pi^{-1}(i+1)$ for $1 \le i \le r$, $F(A) \supset \{1, 2, \ldots, r\}$. □

2.3 Hard problems in the braid group

We describe some of problems in braid groups that are mathematically hard and may be interesting in cryptography.

We say that x and y are *conjugate* if there is an element a such that $y = axa^{-1}$. And for $m < n$, B_m can be considered as a subgroup of B_n generated by $\sigma_1, \ldots, \sigma_{m-1}$.

1. Conjugacy Decision Problem
   Instance: $(x, y) \in B_n \times B_n$.
   Objective: Determine whether x and y are conjugate or not.
2. Conjugacy Search Problem
   Instance: $(x, y) \in B_n \times B_n$ such that x and y are conjugate.
   Objective: Find $a \in B_n$ such that $y = axa^{-1}$.
3. Generalized Conjugacy Search Problem
   Instance: $(x, y) \in B_n \times B_n$ such that $y = bxb^{-1}$ for some $b \in B_m$, $m \leq n$.
   Objective: Find $a \in B_m$ such that $y = axa^{-1}$.
4. Conjugacy Decomposition Problem
   Instance: $(x, y) \in B_n \times B_n$ such that $y = bxb^{-1}$ for some $b \in B_m$, $m < n$.
   Objective: Find $a', a'' \in B_m$ such that $y = a'xa''$.
5. p-th Root Problem
   Instance: $(y, p) \in B_n \times \mathbb{Z}$ such that $y = x^p$ for some $x \in B_n$
   Objective: Find $z \in B_n$ such that $y = z^p$.
6. Cycling[1] Problem
   Instance: $(y, r) \in B_n \times \mathbb{Z}$ such that $y = \mathbf{c}^r(x)$ for some $x \in B_n$
   Objective: Find $z \in B_n$ such that $y = \mathbf{c}^r(z)$.
7. Markov Problem
   Instance: $y \in B_n$ such that y is conjugate to a braid of the form $w\sigma_{n-1}^{\pm 1}$ for some $w \in B_{n-1}$
   Objective: Find $(z, x) \in B_n \times B_{n-1}$ such that $zyz^{-1} = x\sigma_{n-1}^{\pm 1}$.

The Conjugacy Decision Problem and the Conjugacy Search Problem are very important because there are many topologically important problems defined up to conjugacy. But they are so difficult that there is no known polynomial time algorithm to solve this problem. The Generalized Conjugacy Search Problem is a generalized version of the conjugacy problem, which has a restriction on the braid that conjugates x. We will use this problem to propose an one-way function in §3. The Conjugacy Decomposition Problem is trivial for $m = n$ and is easier than Generalized Conjugacy Search Problem. We conjecture that Generalized Conjugacy Search Problem is equivalent to Conjugacy Decomposition Problem for some choices of x. The security of the key agreement scheme proposed in §3 are in fact based on Conjugacy Decomposition Problem.

It seems that one can write down potential one-way functions from the above problems. Furthermore since we can always extract a fixed number of factors at the designated position in a canonical form of a braid, it may be possible to design potential (keyed) hash functions from a combination of the above problems.

Finally we remark that the Markov Problem is closely related to the study of knots and links via braids. As every knot theorist dreams of a complete classification of knots and links, this problem should be hard.

[1] For an n-braid $x = \Delta^u A_1 \cdots A_p$ in the left-canonical form, the *cycling* of x is defined by $\mathbf{c}(x) = \Delta^u A_2 \cdots A_p \tau^u(A_1)$, where the automorphism $\tau \colon B_n \to B_n$ is defined by $\tau(\sigma_i) = \sigma_{n-i}$ for $i = 1, \ldots, n-1$.

3 The cryptosystem using braid groups

In this section, we propose a one-way function based on the difficulty of the Generalized Conjugacy Search Problem. Also we propose a key agreement scheme and a PKC using the proposed one-way function. But we don't have a digital signature scheme based on the braid groups yet.

Note: All the braids in this section are supposed to be in the left-canonical form. For example, for $a, b \in B_n$, ab means the left-canonical form of ab and so it is hard to guess its factors a or b from ab.

3.1 Proposed one-way function

We consider two subgroups LB_ℓ and RB_r of $B_{\ell+r}$. LB_ℓ (resp. RB_r) is the subgroup of $B_{\ell+r}$ consisting of braids made by braiding left ℓ strands(resp. right r strands) among $\ell+r$ strands. Thus LB_ℓ is generated by $\sigma_1, \ldots, \sigma_{\ell-1}$ and RB_r is generated by $\sigma_{\ell+1}, \ldots, \sigma_{\ell+r-1}$. An important role is played by the commutative property that for any $a \in LB_\ell$ and $b \in RB_r$, $ab = ba$.

Now we propose the following one-way function

$$f\colon LB_\ell \times B_{\ell+r} \to B_{\ell+r} \times B_{\ell+r}, \qquad f(a, x) = (axa^{-1}, x).$$

It is a one-way function because given a pair (a, x), it is easy to compute axa^{-1} but all the known attacks need exponential time to compute a from the data (axa^{-1}, x). This one-way function is precisely based on the Generalized Conjugacy Search Problem.

The securities of our key agreement scheme and PKC are based on the difficulty of the following problem.

[Base Problem]
Instance: The triple (x, y_1, y_2) of elements in $B_{\ell+r}$ such that $y_1 = axa^{-1}$ and $y_2 = bxb^{-1}$ for some hidden $a \in LB_\ell$ and $b \in RB_r$.
Objective: Find $by_1b^{-1} (= ay_2a^{-1} = abxa^{-1}b^{-1})$.

We do not know whether this problem is equivalent to the Generalized Conjugacy Search Problem, even though the latter problem implies the former problem. The two problems seem to have the almost same complexity and this phenomenon is similar to the case of the Diffie-Hellman problem and the discrete logarithm problem.

The role of x is similar to that of g in the Diffie-Hellman problem to find g^{xy} from g^x and g^y. In order to make our base problem hard, x must be sufficiently complicated by avoiding the "reducible" braids x_1x_2z where $x_1 \in LB_\ell$, $x_2 \in RB_r$ and z is a $(\ell + r)$-braid that commutes with both LB_ℓ and RB_r as depicted in Figure 4 for $\ell = r = 3$. If x were decomposed into x_1x_2z, then $by_1b^{-1} = (ax_1a^{-1})(bx_2b^{-1})z$ would be obtained from $y_1 = (ax_1a^{-1})x_2z$ and $y_2 = x_1(bx_2b^{-1})z$ without knowing a and b. It is shown by Fenn-Rolfsen-Zhu in [15] that $(\ell + r)$-braids that commute with RB_r (or LB_ℓ) are of the form x_1z (or x_2z, respectively) up to full twists Δ_ℓ^2 and Δ_r^2 of left ℓ strands and right r

Fig. 4. An example of reducible braid

strands. The probability for a randomly chosen $(\ell + r)$-braid of canonical length q to be reducible is small, that is, roughly $(\ell!r!/(\ell + r)!)^q$.

Conjugacy Decomposition Problem also implies the base problem since for $a', a'' \in LB_\ell$ such that $a'xa'' = y_1$, we have $a'y_2a'' = a'bxb^{-1}a'' = ba'xa''b^{-1} = by_1b^{-1}$. We note that Conjugacy Decomposition Problem is trivial if x is reducible. Thus a necessary condition on x for which Conjugacy Decomposition Problem becomes equivalent to Generalized Conjugacy Search Problem is that x is not reducible. A sufficient condition on x is that xcx^{-1} is not in B_m for each nontrivial $c \in B_m$. But we do not have a good characterization of this sufficient condition yet and further study on the choice of x may be required to maintain the soundness of our base problem.

Recall the surjection $\rho\colon B_n \to \Sigma_n$ into the permutation group. In order to prevent adversaries from computing $\rho(a)$ and $\rho(b)$ by looking at $\rho(x)$, $\rho(y_1)$ and $\rho(y_2)$, x should be a pure braid so that $\rho(y_1)$ and $\rho(y_2)$ as well as $\rho(x)$ are the identity.

3.2 Key agreement

Now we propose a key agreement system between A(lice) and B(ob). This is the braid group version of the Diffie-Hellman key agreement system.

1. **Preparation step:** An appropriate pair of integers (ℓ, r) and a sufficiently complicated $(\ell + r)$-braid $x \in B_{\ell+r}$ are selected and published. The requirement to be sufficiently complicated has been discussed in §3.1.
2. **Key agreement:** Perform the following steps each time a shared key is required.
   (a) A chooses a random secret braid $a \in LB_\ell$ and sends $y_1 = axa^{-1}$ to B.
   (b) B chooses a random secret braid $b \in RB_r$ and sends $y_2 = bxb^{-1}$ to A.
   (c) A receives y_2 and computes the shared key $K = ay_2a^{-1}$.
   (d) B receives y_1 and computes the shared key $K = by_1b^{-1}$.

Since $a \in LB_\ell$ and $b \in RB_r$, $ab = ba$. Thus

$$ay_2a^{-1} = a(bxb^{-1})a^{-1} = b(axa^{-1})b^{-1} = by_1b^{-1}$$

and so Alice and Bob obtain the same braid.

Since the Anshel-Anshel-Goldfeld's key agreement in [2] is also based on combinatorial groups and conjugacy problems, it seems necessary to discuss the difference between our key agreement and the Anshel-Anshel-Goldfeld's key agreement. The points of their algebraic key establishment protocol in [2] are the homomorphic property of an one-way function and the public key of multiple arguments. But our key agreement relies neither on the homomorphic property nor on the public key of multiple arguments.

The group theoretic application in [2] uses the following generalization of the usual conjugacy search problem:

> Given words $t_1, \ldots, t_k$, and $at_1a^{-1}, \ldots, at_ka^{-1}$ in a group G, find such a word a in G.

On the other hand our key agreement is based on another generalization of the conjugacy search problem as follows:

> Given words x and axa^{-1} in a group G and given a subgroup H of G, find such a word a in H.

We believe that the two generalizations are independent each other, especially for the braid group B_n and its subgroup LB_ℓ under our current mathematical knowledge.

It should be also noted that the trapdoors of two key agreements are distinct. Our scheme uses the commutativity between two subgroups LB_ℓ and RB_r. On the other hand the Anshel-Anshel-Goldfeld's scheme uses the homomorphic property of conjugations, that is, $(asa^{-1})(ata^{-1}) = asta^{-1}$.

3.3 Public-key cryptosystem

By using the key agreement system in §3.2, we construct a new PKC. Let $H\colon B_{\ell+r} \to \{0,1\}^k$ be an ideal hash function from the braid group to the message space.

1. **Key generation:**
   (a) Choose a sufficiently complicated $(\ell + r)$-braid $x \in B_{\ell+r}$.
   (b) Choose a braid $a \in LB_\ell$.
   (c) Public key is (x, y), where $y = axa^{-1}$; Private key is a.
2. **Encryption:** Given a message $m \in \{0,1\}^k$ and the public key (x, y).
   (a) Choose a braid $b \in RB_r$ at random.
   (b) Ciphertext is (c, d), where $c = bxb^{-1}$ and $d = H(byb^{-1}) \oplus m$.
3. **Decryption:** Given a ciphertext (c, d) and private key a, compute $m = H(aca^{-1}) \oplus d$.

Since a and b commute, $aca^{-1} = abxb^{-1}a^{-1} = baxa^{-1}b^{-1} = byb^{-1}$. So $H(aca^{-1}) \oplus d = H(byb^{-1}) \oplus H(byb^{-1}) \oplus m = m$ and the decryption recovers the original braid m.

Because H is an ideal hash function, our PKC is semantically secure relative to the decisional version of our base problem; if the adversary can compute some information of the message from the public key and the ciphertext, (s)he can also compute some information of $byb^{-1} = abxa^{-1}b^{-1}$ from x, axa^{-1} and bxb^{-1}.

We hope that one can make a semantically secure PKC using the proposed one-way function without assuming the hash function to be an ideal hash function.

4 The theoretic operating characteristics and implementation

In this section, we discuss the theoretic operating characteristics of our PKC and the security/message length parameters for future implementations. Because our PKC has not been fully implemented yet as a computer program, we can not compare its speed with other PKC's. But we can report the speed of a conversion algorithm into left canonical forms that is the essential part of our PKC

Recall that our PKC uses three braids $x \in B_{\ell+r}$, $a \in LB_\ell$ and $b \in RB_r$, and the ciphertext is $(bxb^{-1}, H(abxa^{-1}b^{-1}) \oplus m)$. When we work with braids, we should consider two parameters, the braid index and the canonical length. For simplicity, we assume that the braid indexes in our PKC are $\ell = r = \frac{n}{2}$ and the canonical lengths are $\text{len}(x) = \text{len}(a) = \text{len}(b) = p$. The followings are the discussions about the operating characteristics of our PKC, which is summarized in Table 1.

1. An n-permutation can be represented by an integer $0 \le N < n!$. Since $n! \sim \exp(n \log n)$, a braid with p canonical factors can be represented by a bit string of size $pn \log n$.
2. For braids $y_1, y_2 \in B_n$, $\text{len}(y_1 y_2) \le \text{len}(y_1) + \text{len}(y_2)$ and for $y_1 \in LB_\ell, y_2 \in RB_r$, $\text{len}(y_1 y_2) = \max\{\text{len}(y_1), \text{len}(y_2)\}$. So $\text{len}(bxb^{-1})$ and $\text{len}(abxa^{-1}b^{-1})$ are at most $3p$. For generic choices of a, b, and x, they are no less than $2p$. Thus we assume that $\text{len}(bxb^{-1})$ and $\text{len}(abxa^{-1}b^{-1})$ are between $2p$ and $3p$.
3. The size of the private key a is $p\ell \log \ell \sim p\frac{n}{2} \log \frac{n}{2} \sim \frac{1}{2} pn \log n$.
4. The size of the public key bxb^{-1} is at most $3pn \log n$.
5. By Theorem 3, the number of n braids with $2p$ canonical factors is at least the exponential of

$$\log \left(\left\lfloor \tfrac{n-1}{2} \right\rfloor ! \right)^{2p} = 2p \log \left(\left\lfloor \tfrac{n-1}{2} \right\rfloor ! \right) \sim 2p \log \left(\tfrac{n}{2}! \right) \sim 2p \cdot \tfrac{n}{2} \cdot \log \tfrac{n}{2} \sim pn \log n.$$

   Thus we may let the bit length of $H(abxa^{-1}b^{-1})$ equal $pn \log n$ and so the message length is also $pn \log n$. Since the bit size of bxb^{-1} is at most $3pn \log n$, the size of ciphertext is at most $3pn \log n + pn \log n = 4pn \log n$. Hence the message expansion is less than 4-1.
6. As noted earlier, the encryption/decryption speed is $\mathcal{O}(p^2 n \log n)$.
7. The hardness of the brute force attack to compute a from axa^{-1}, or equivalently to compute b from bxb^{-1}, is proportional to $(\ell!)^p = (\frac{n}{2}!)^p \sim \exp(\frac{1}{2} pn \log n)$.

Plaintext block	$pn \log n$ bits
Ciphertext block	$4pn \log n$ bits
Encryption speed	$\mathcal{O}(p^2 n \log n)$ operation
Decryption speed	$\mathcal{O}(p^2 n \log n)$ operation
Message expansion	4-1
Private key length	$\frac{1}{2} pn \log n$ bits
Public key length	$3pn \log n$ bits
Hardness of brute force attack	$(\frac{n}{2}!)^p \sim \exp(\frac{1}{2} pn \log n)$

Table 1. The operating characteristics of our PKC

Both the security level and the message length are affected at the same extent by p and $n \log n$, but the speed is quadratic in p and linear in $n \log n$. Thus it is better to increase n rather than p in order to increase the security level.

Table 2 shows speed of the canonical form algorithm in the braid group. It is the total ellapsed time of our C-program which takes a pair of integers (n, s), and then generates s random canonical factors $A_1, \ldots, A_s$, and then computes the left-canonical form of $A_1 \cdots A_s$ in Pentium III 450MHz. The table shows that the multiplication in braid groups is efficient.

The C-program inputs pairs (n, s) and outputs the ellasped time t. The other entries in Table 2 are computed from (n, s) and t as follows:

1. Let $p = \lceil \frac{s}{3} \rceil + 1$ be the canonical length of the fixed braid x of our PKC. Let $q = \frac{n-p}{2}$. Then we consider the PKC, where $\mathrm{len}(x) = p$, $\mathrm{len}(a) = \mathrm{len}(b) = q$.
2. The expected message length k is computed by $p \log n!$.
3. Kbits/(sec) is computed by $k/(2^{10} t)$.
4. Hardness of brute force attack is computed by $\log(\frac{n}{2}!)^q$.

s	p	Braid index n	Expected message length k	Ellapsed time t (sec)	Kbits/(sec)	Hardness of brute force attack
11	5	50	1071	0.0112	93.4	251
11	5	70	1662	0.0210	77.3	399
11	5	90	2294	0.0344	65.2	559
17	7	50	1499	0.0173	84.6	418
17	7	70	2327	0.0325	69.9	665
17	7	90	3212	0.0532	59.0	931
32	12	50	2570	0.0326	77.0	837
32	12	70	3989	0.0611	63.8	1329
32	12	90	5507	0.1037	51.9	1863

Table 2. The performance of the algorithm converting into the left canonical forms.

5 Security analysis

In this section, we analyze the security of the proposed encryption scheme.

5.1 Similarity with ElGamal scheme

Our PKC is similar to the ElGamal PKC in design and it has the following properties.

1. The problem of breaking our PKC is equivalent to solving the base problem, as breaking the ElGamal PKC is equivalent to solving the Diffie-Hellman problem. In the proposed scheme, the ciphertext is

$$(c, d) = (bxb^{-1}, H(abxa^{-1}b^{-1}) \oplus m)$$

   and decrypting the ciphertext into a plaintext m is equivalent to knowing $abxa^{-1}b^{-1}$.
2. Like any other probabilistic PKC's, it is critical to use different key b for each session: If the same session key b is used to encrypt both of m_1 and m_2 whose corresponding ciphertexts are (c_1, d_1) and (c_2, d_2), then m_2 can be easily computed from (m_1, d_1, d_2) because $H(byb^{-1}) = m_1 \oplus d_1 = m_2 \oplus d_2$.

5.2 Brute force attack

A possible brute force attack is to compute a from axa^{-1} or b from bxb^{-1}, which is just an attack to Generalized Conjugacy Search Problem. Assume that we are given a pair (x, y) of braids in $B_{\ell+r}$, such that $y = axa^{-1}$ for some $a \in LB_\ell$. The braid a can be chosen from an infinite group LB_ℓ in theory. But in a practical system, the adversary can generate all braids $a = \Delta^u A_1 \dots A_p$ in the canonical form with some reasonable bound for p and check whether $y = axa^{-1}$ holds. The necessary number is at least $(\frac{\ell-1}{2}!)^p$ by Theorem 3. If $\ell = 45$ and $p = 2$, then $(\frac{\ell-1}{2}!)^p > 2^{139}$, which shows that the brute force attack is of no use.

We note that there might be another $a' \in LB_\ell$ such that $y = a'xa'^{-1}$. Then $a^{-1}a'$ must be a member of the centralizer $C(x)$ of x. For a generic x and a fixed canonical length, the probability for a braid in LB_ℓ to be a member of $C(x) \cap LB_\ell$ seems negligible, that is, it is hard to find such an a' different from a.

Another possible brute force attack is to find $a' \in LB_\ell$ such that $x^{-1}a^{-1}a'x \in LB_\ell$, which is an attack to Conjugacy Decomposition Problem. As we conjectured earlier, there are choices of x so that Conjugacy Decomposition Problem implies Generalized Conjugacy Search Problem, that is, such an a' must equals to a. Thus we need to concern only about an attack to Generalized Conjugacy Search Problem.

5.3 Attack using the super summit set

The adversary may try to use a mathematical solution to the conjugacy problem by Garside [16], Thurston [14], Elrifai-Morton [13] and Birman-Ko-Lee [6]. But the known algorithms find an element $a \in B_{\ell+r}$, not in LB_ℓ. Hence the attack using the super summit set will not be successful.

6 Further study

1. We think that further primitives and cryptosystems can be found by using hard problems in the braid groups. For example, a new digital signature scheme is waiting for our challenge.
2. Generalized Conjugacy Search Problem implies Conjugacy Decomposition Problem that in turn implies the base problem in §3.1. We would like to know what choice of x makes these three problems equivalent. If this question is too challenging, it is nice to know a practical sufficient condition on x that makes the first two problems equivalent. It seems hard to prove directly that Conjugacy Decomposition Problem or the base problem are intractible. On the other hand, Generalized Conjugacy Search Problem seems mathematically more interesting and so it could attract more research.
3. We may try to use other groups with an one-way function based on the conjugacy problem and so on. To support our ideas, the group must have the following properties.
   - The word problem should be solved by a fast algorithm. It would be much better if the word problem is solved by a fast algorithm which computes a canonical form. For example, the automatic groups may be good candidates [14, 25].
   - The conjugacy problem must be hard. The permutation group does not satisfy this requirement.
   - It should be easy to digitize the group element.

   Our idea can be applied to matrix groups. In particular, for an n-braid, we can compute its image of Burau representation [5], which is an $n \times n$ matrix in $GL_n(\mathbb{Z}[t, t^{-1}])$. One might expect that the conjugacy problem in this matrix group is easier than in the braid groups. But it does not seem so. And it is not easy to encode the message into a matrix and vice versa.

Acknowledgement

We wish to thank Dan Boneh, Mike Brenner and anonymous referees for their valuable comments and suggestions. The first two authors also wish to thank Joan Birman for her kind introduction of the braid theory and her continuing encouragement. The first author was supported in part by the National Research Laboratory Grant from the Ministry of Science and Technology in the program year of 1999.

References

1. I. Anshel and M. Anshel, *From the Post-Markov theorem through decision problems to public-key cryptography*, Amer. Math. Monthly **100** (1993), no. 9, 835–844.
2. I. Anshel, M. Anshel and D. Goldfeld, *An algebraic method for public-key cryptography*, Mathematical Research Letters **6** (1999) 287–291.
3. E. Artin, *Theory of braids*, Annals of Math. **48** (1947), 101–126.
4. C. H. Bennet and G. Brassard, *Quantum cryptography: Public key distribution and coin tossing*, Proc. IEEE Int. Conf. Computers, Systems and Signal Processing (Bangalore, India, 1984), 175–179.
5. J. S. Birman, *Braids, links and mapping class groups*, Annals of Math. Study, no. **82**, Princeton University Press (1974).
6. J. S. Birman, K. H. Ko and S. J. Lee, *A new approach to the word and conjugacy problems in the braid groups*, Advances in Math. **139** (1998), 322-353.
7. D. Boneh, *Twenty years of attacks on the RSA cryptosystem*, Notices Amer. Math. Soc. **46** (1999), 203–213.
8. G. Brassard, *A note on the complexity of cryptography*, IEEE Transactions on Information Theory **25** (1979), 232-233.
9. E. F. Brickell, *Breaking iterated knapsacks*, Advances in Cryptology, Proceedings of Crypto '84, Lecture Notes in Computer Science **196**, ed. G. R. Blakley and D. Chaum, Springer-Verlag (1985), 342–358.
10. P. Dehornoy, *A fast method for comparing braids*, Advances in Math. **125** (1997), 200-235.
11. W. Diffie and M. E. Hellman, *New directions in cryptography*, IEEE Transactions on Informaton Theory **22** (1976), 644–654.
12. T. ElGamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Transactions on Information Theory **31** (1985), 469–472.
13. E. A. Elrifai and H. R. Morton, *Algorithms for positive braids*, Quart. J. Math. Oxford **45** (1994), no. 2, 479–497.
14. D. Epstein, J. Cannon, D. Holt, S. Levy, M. Paterson and W. Thurston, *Word processing in groups*, Jones & Bartlett, 1992.
15. R. Fenn, D. Rolfsen and J. Zhu *Centralisers in the braid group and singular braid monoid*, Enseign. Math. (2) **42** (1996), no. 1-2, 75–96.
16. F. A. Garside, *The braid group and other groups*, Quart. J. Math. Oxford **20** (1969), no. 78, 235–254.
17. M. Garzon and Y. Zalcstein, *The complexity of Grigorchuk groups with application to cryptography*, Theoretical Computer Sciences **88** (1991) 83–98.
18. O. Goldreich, S. Goldwasser and S. Halevi, *Public-key cryptosystems from lattice reduction problems*, Advances in Cryptology, Proceedings of Crypto '97, Lecture Notes in Computer Science **1294**, ed. B. Kaliski, Springer-Verlag (1997), 112–131.
19. E. S. Kang, K. H. Ko and S. J. Lee, *Band-generator presentation for the 4-braid group*, Topology Appl. **78** (1997), 39-60.
20. K. Komaya, U. Maurer, T. Okamoto and S. Vanston, *New public-key schemes bases on elliptic curves over the ring* $\mathbf{Z}_n$, Advances in Cryptology, Proceedings of Crypto '91, Lecture Notes in Computer Science **576**, ed. J. Feigenbaum, Springer-Verlag (1992), 252–266
21. N. Koblitz, *Algebraic aspects of cryptography*, Algorithms and Computations in Mathematics **3** (1998) Springer-Verlag, Berlin.
22. J. C. Lagarias, *Knapsack public key cryptosystems and Diophantine approximation*, Advances in Cryptology: Proceedings of Crypto '83, ed. by D. Chaum, Plenum Publishing (1984), 3–24.

23. K. McCurley, *A key distribution system equivalent to factoring*, Journal of Cryptology **1** (1988), 95–105.
24. R. C. Merkle and M. E. Hellman, *Hiding information and signatures in trapdoor knapsacks*, IEEE Transactions on Information Theory **24** (1978), 525–530.
25. L. Mosher, *Mapping class groups are automatic*, Ann. Math. **142** (1995), 303–384.
26. A. M. Odlyzko, *The rise and fall of knapsack cryptosystems*, Cryptology and Computational Number Theory, Proc. Symp. App. Math. **42** (1990), 75–88.
27. M. S. Paterson and A. A. Rasborov, *The set of minimal braids is co-NP-complete*, J. Algorithms. **12** (1991), 393–408.
28. M. O. Rabin, *Digitized signatures and public-key functions as intractible as factorization*, MIT Laboratory for Computer Science Technical Report, LCS/TR-212 (1979).
29. R. L. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, Communications of the ACM **21** (1978), 120–126.
30. A. Shamir, *A polynomial time algorithm for breaking the basis Merkle-Hellman cryptosystem*, Advances in Cryptology: Proceedings of Crypto '82, ed. by D. Chaum et al., Plenum Publishing (1983), 279–288.
31. R. Siromoney and L. Mathew, *A public key cryptosystem based on Lyndon words*, Information Proceeding Letters **35** (1990) 33-36.
32. P. Smith and M. Lennon, *LUC: A new public key system*, Proceedings of the IFIP TC11 Ninth International Conference on Information Security, ed. E. Dougall, IFIP/Sec 93, 103–117, North-Holland, 1993.
33. R. P. Stanley, *Enumerative combinatorics*, Wadsworth and Brooks/Cole, 1986.
34. Y. Tsiounis and M. Yung, *On the security of Elgamal based encryption*, In PKC '98, Lecture Notes in Computer Science **1431**, Springer-Verlag (1998), 117–134.
35. S. Vaudenay, *Cryptanalysis of the Chor-Rivest Cryptosystem*, Advances in Cryptology: Proceedings of Crypto '98, Lecture Notes in Computer Science **1462**, ed. Krawczyk, Springer-Verlag (1998), 243–256.
36. H. Williams, *Some public-key crypto-funtions as intractible as factorization*, Advances in Cryptology, Proceedings of Crypto '84, Lecture Notes in Computer Science **196**, ed. G. R. Blakley and D. Chaum, Springer-Verlag (1985), 66–70.
37. N. R. Wagner and M. R. Magyarik, *A public-key cryptosystem based on the word problem*, Advances in Cryptology, Proceedings of Crypto '84, Lecture Notes in Computer Science **196**, ed. G. R. Blakley and D. Chaum, Springer-Verlag (1985), 19–36.