Pseudorandomness from Braid Groups

Eonkyung Lee, Sang Jin Lee, and Sang Geun Hahn

Department of Mathematics,
Korea Advanced Institute of Science and Technology,
Taejon 305-701, Republic of Korea
{eklee,sjlee,sghahn}@mathx.kaist.ac.kr

Abstract. Recently the braid groups were introduced as a new source for cryptography. The group operations are performed efficiently and the features are quite different from those of other cryptographically popular groups. As the first step to put the braid groups into the area of pseudorandomness, this article presents some cryptographic primitives under two related assumptions in braid groups. First, assuming that the conjugacy problem is a one-way function, say f, we show which particular bit of the argument x is pseudorandom given $f(x)$. Next, under the decision Ko-Lee assumption, we construct two provably secure pseudorandom schemes: a pseudorandom generator and a pseudorandom synthesizer.

1 Introduction

The notions of pseudorandomness and onewayness which are closely related are quite important in modern cryptography [8,1,17,12]. These concepts are informally stated as: (i) A distribution is *pseudorandom* if no efficient algorithm can distinguish it from the uniform distribution [26]. (ii) A function is *one-way* if it is easy to evaluate but hard to invert [9].

Recently, some mathematically hard problems in braid groups have been proposed as new candidates for cryptographic one-way functions [2,19]. A braid group B_n is an infinite non-commutative group naturally arising from geometric braids composed of n strands. One of the famous problems in braid groups is the *conjugacy problem*: Given $(\alpha, \beta) \in B_n \times B_n$, find (*or* determine whether there exists) $\chi \in B_n$ such that $\beta = \chi^{-1}\alpha\chi$. This problem was first introduced in the 1920s, and no polynomial-time algorithm is known for $n \geq 5$. A variant of this problem was first applied to cryptography to build a key agreement scheme by Anshel *et al.* [2].

Ko *et al.* [19] introduced another variant of this problem: Given $\alpha, \chi_1^{-1}\alpha\chi_1, \chi_2^{-1}\alpha\chi_2 \in B_n$, where χ_1 and χ_2 are contained in some known subgroups of B_n so that $\chi_1\chi_2 = \chi_2\chi_1$, find $\chi_2^{-1}\chi_1^{-1}\alpha\chi_1\chi_2 \in B_n$. For convenience, we call this problem the *Ko-Lee problem.* The Ko-Lee problem looks like the Diffie-Hellman problem in their structures, but it does not in their internal properties because of the different characteristics of the braid groups from finite commutative groups. For instance, a braid group is non-commutative and it has no finite subgroup except for the trivial subgroup. As the basis of the Ko-Lee problem, they introduced, by restricting the conjugacy problem to a smaller braid group, the (n, m)-*generalized conjugacy problem* (GCP): Given $(\alpha, \beta) \in B_n \times B_n$ and $m(\leq n)$, find $\chi \in B_m$ such that $\beta = \chi^{-1}\alpha\chi$. Like the conjugacy problem, the GCP and the Ko-Lee problem have no polynomial-time solving algorithm yet.

The motivation for this article is that the braid groups have potential for a good source to enrich cryptography from the point of view of their features and efficient operations. In the sequel to key agreement schemes [2,19] and a public-key cryptosystem [19], we discuss how to construct cryptographic primitives in the area of pseudorandomness from the two related assumptions in braid groups: the intractability assumptions of the conjugacy and the Ko-Lee problems. We call the latter the *Ko-Lee assumption* (KL-Assumption).

1.1 The Ko-Lee Problem

As a basic pseudorandom primitive, a pseudorandom generator is informally defined to be an efficient algorithm expanding short random bit sequences into long pseudorandom bit sequences [26,8].

Naor *et al.* [23] first introduced the notion of pseudorandom synthesizer as a stronger one than pseudorandom generator in the following sense: While a pseudorandom generator, G, guarantees the pseudorandomness of $\{G(z_i)\}_{1\leq i\leq n}$ only when $z_1, \ldots, z_n$ are chosen uniformly and *independently*, a pseudorandom synthesizer, S, guarantees the pseudorandomness of $\{S(z_i)\}_{1\leq i\leq n}$ even when the z_i's are not *completely independent*. Loosely speaking, a pseudorandom synthesizer is a two variable function $S(\cdot, \cdot)$, so that if polynomially many random assignments are chosen to both variables, $(x_1, \ldots, x_m)$ and $(y_1, \ldots, y_m)$, then the output of S on all the combinations of these assignments, $(S(x_i, y_j))_{1\leq i,j\leq m}$, is pseudorandom.

Our Result: From the KL-Assumption, we formally derive a decisional version mentioned to refer to the security of the braid public-key cryptosystem [19]. Under the decision Ko-Lee assumption (DKL-Assumption), we construct a pseudorandom generator and a pseudorandom synthesizer and show that they are provably secure.

1.2 The Conjugacy Problem

The Ko-Lee problem was originally proposed as a variant of the conjugacy problem to induce a trapdoor one-way function (for a public-key cryptosystem). However, it looks easier to solve than the conjugacy problem. Since pseudorandomness needs no trapdoor, the conjugacy problem itself can be considered.

If f is a one-way function, every bit of the argument x cannot be easily computed from $f(x)$. A natural question is whether there is a specific bit of x which is not distinguished from a random bit by any efficient algorithm given $f(x)$. This question was first addressed by Blum *et al.* [8]. Demonstrating such a pseudorandom bit for the discrete exponentiation function, they introduced the notion of hard-core predicate as a cryptographically useful tool. Loosely speaking, a *hard-core predicate* b of a function f is a polynomial-time computable boolean predicate such that $b(x)$ is hard to predict from $f(x)$. So far, two kinds of hard-core predicates have been proposed. On the one hand, for a few one-way function f's, there has been discovered a particular bit of x, the so-called *hard-core bit*, which is the source of $b(x)$ by the unique characteristic of f [8,1]. For instance, Alexi *et al.* [1] showed that $b(x)$ points to the least significant bit of x for the RSA and the Rabin functions. On the other hand, for any one-way function, one can make a hard-core predicate by Goldreich-Levin's construction [14]. More precisely, for any one-way function f, the inner-product mod 2 of x and r is a hard-core of $g(x,r) \overset{\text{def}}{=} (f(x), r)$. To distinguish these two kinds of hard-core predicates, we call the former kind the *peculiar* one and the latter kind the *generic* one.

Considering that among a number of known one-way functions only the RSA, the Rabin, and the discrete exponentiation functions have their peculiar hard-core predicates, it is interesting to find it for the conjugacy problem. It indicates which bit of the solution is equally difficult to compute as the entire solution.

The conjugacy problem in braid groups is quite different from those above one-way functions in the sense that it is not a group homomorphism. Since such a property is the basis for the construction of the previous peculiar hard-core predicates, we should take a completely different way to construct a peculiar hard-core for the conjugacy problem.

Our Result: We first present a collection of one-way functions, CNJ, under the intractability assumption of the $(n, n-1)$-GCP, which is almost the conjugacy problem from a computational complexity point of view. And we present *two* hard-core bits of CNJ. Using one of them, we construct a peculiar hard-core predicate, INF, and prove that predicting $\text{INF}(x)$ from $\text{CNJ}(x)$ is as hard as inverting $\text{CNJ}(x)$. Likewise the other hard-core bit.

1.3 Outline

In §2, we introduce some notations and briefly describe the braid groups. In §3, we examine the bit security in the conjugacy problem (§3.1), present a collection of one-way functions based on that problem (§3.2), and construct a hard-core predicate of the one-way function (§3.3). In §4, we construct a pseudorandom generator (§4.1) and a pseudorandom synthesizer (§4.2).

2 Preliminaries

2.1 Notations

Basic notation: Let **N** and **Z** denote the set of all natural numbers and the set of all integers, respectively. For any bit-string x, $||x||$ denotes its length (i.e. the number of bits in x). For a finite set S, $|S|$ denotes the cardinality of S and $||S||$ denotes the maximum among the bit-lengths of elements of S. The notation $(a_{i,j})_{1\le i\le n, 1\le j\le m}$ denotes an $(n \times m)$-matrix whose (i,j)-entry is $a_{i,j}$.

Probability notation: The following notations are based on [16,15,3].

A probability distribution $\mathcal{D}$ on a finite set S assigns a probability $\mathcal{D}(s) \geq 0$ to each $s \in S$, and thus $\sum_{s \in S} \mathcal{D}(s) = 1$. For a distribution $\mathcal{D}$, $[\mathcal{D}]$ denotes the support of $\mathcal{D}$ (the set of elements of positive probability). If a random variable x is distributed according to $\mathcal{D}$ on S, we write $x \overset{\mathcal{D}}{\leftarrow} S$, or simply $x \leftarrow \mathcal{D}$ if the set S is obvious from the context. The notation $x_1, \dots, x_n \leftarrow \mathcal{D}$ indicates that n random variables $x_1, \dots, x_n$ are independently distributed according to $\mathcal{D}$ on S.

If f is a function mapping S to a set T, then $\langle f(x) : x \leftarrow \mathcal{D} \rangle$ is a random variable that defines a distribution $\mathcal{E}$, where for all $t \in T$, $\mathcal{E}(t) = \sum_{s \in S, f(s)=t} \mathcal{D}(s)$.

If $\mathcal{A}$ is a probabilistic algorithm, then for any input $x, y, \dots$ the notation $\mathcal{A}(x, y, \dots)$ refers to the probability distribution induced by its internal random coin tosses. So if $x \leftarrow \mathcal{D}, y \leftarrow \mathcal{E}, \dots$ are random variables, then $\langle \mathcal{A}(x, y, \dots) : x \leftarrow \mathcal{D}; y \leftarrow \mathcal{E}; \dots \rangle$ represents the random variable distributed according to $\mathcal{D}, \mathcal{E}, \dots$ and its internal random coin tosses.

We let $x \overset{u}{\leftarrow} S$ indicate that x is uniformly distributed on S; i.e., for all $s \in S$, $\Pr[x = s : x \overset{u}{\leftarrow} S] = 1/|S|$.

For probability distributions $\mathcal{D}, \mathcal{E}, \dots$, the notation $\Pr[p(x, y, \dots) : x \leftarrow \mathcal{D}; y \leftarrow \mathcal{E}; \cdots]$ denotes the probability that the predicate $p(x, y, \dots)$ is true after the (ordered) execution of the algorithms $x \leftarrow \mathcal{D}, y \leftarrow \mathcal{E}$, etc..

PPTA is short for "probabilistic polynomial time algorithm in its input length(s)".

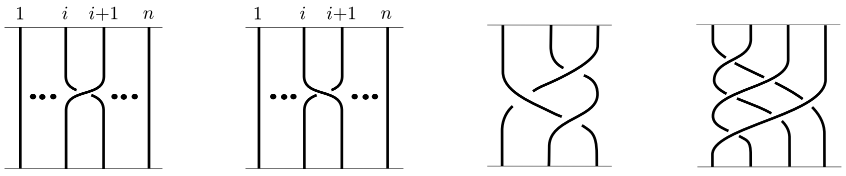
2.2 The Braid Groups

In this section, we briefly review some basic material for braid groups. See [6, 10,7] for details. For each integer $n \geq 2$, the n-braid group B_n is defined by the following group presentation

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

The integer n is called the *braid index* and each element of B_n is called an *n-braid*. An n-braid has the following geometric interpretation: it is a set of disjoint n strands which run essentially to the same direction (our convention is vertical direction). The multiplication $\alpha\beta$ of two braids α and β is the braid obtained by positioning α on the top of β, the identity e_n is the braid consisting of n straight vertical strands, and the inverse of α is the reflection of α with respect to a horizontal plane. Examples are given in Figure 1 (a,b,c). Henceforth, let σ_i denote only a generator of the corresponding braid group.

B_n^+ denotes the monoid defined by the generators and relations in the above presentation, and its elements are called *positive n-braids*. To each permutation $\pi = b_1 b_2 \cdots b_n$, we associate a positive n-braid obtained by connecting the upper i-th point to the lower b_i-th point by a straight line. Such braids as this are called *permutation braids* or *canonical factors*. The permutation n-braid corresponding to the permutation $(n)(n-1)\cdots(2)(1)$ is called the *fundamental braid* and denoted by Δ_n. See Figure 1 (d) for example. For $\alpha \in B_n^+$, define two sets $S(\alpha) = \{i \mid \alpha = \sigma_i \beta \text{ for some } \beta \in B_n^+\}$ and $F(\alpha) = \{i \mid \alpha = \beta \sigma_i \text{ for some } \beta \in B_n^+\}$.



(a) σ_i (b) σ_i^{-1} (c) $\sigma_2\sigma_1^{-1}\sigma_2$ (d) Δ_4

Fig. 1. An example of braids

Every braid $\chi \in B_n$ has a unique decomposition called the *left-canonical form*, $\chi = \Delta_n^u \chi_1 \cdots \chi_k$, where $u \in \mathbf{Z}$ and χ_i's are permutation braids except for e_n and Δ_n such that $F(\chi_i) \supset S(\chi_{i+1})$. In this article, all the braids are supposed to be in their left-canonical forms. Hence, for $\alpha, \beta \in B_n$, $\alpha\beta$ means the left-canonical form of $\alpha\beta$ and so it is hard to guess its original factor α or β from $\alpha\beta$.

For $m < n$, B_m is regarded as the subgroup of B_n generated only by $\sigma_1, \ldots, \sigma_{m-1}$ of B_n, and so $\Delta_m (\in B_n)$ is a permutation n-braid corresponding to a permutation $(m)(m-1)\cdots(2)(1)(m+1)(m+2)\cdots(n)$.

Due to [10,7], braid groups with all their operations—multiplication, inversion, converting into left-canonical forms—are efficiently handled by computers.

3 Hard-Core Predicate

From the intractability assumption of the conjugacy problem, one can naturally derive a one-way function, $\textsc{cnj}_\alpha : B_n \longrightarrow B_n$, defined by $\textsc{cnj}_\alpha(\chi) = \chi^{-1}\alpha\chi$, where $\alpha \in B_n$.

Our goal in this section is to construct a *peculiar* hard-core predicate of $\textsc{cnj}_\alpha$. Therefore, we should discover for $\textsc{cnj}_\alpha$ the hard-core bit of a braid into which the one-wayness of $\textsc{cnj}_\alpha$ is transformed.

Notice that we are in different situation from previous ones for the following reasons: (i) A braid is not naturally expressed as a digit. (ii) $\textsc{cnj}_\alpha$ is not a group homomorphism. By (i), we should find a different type of bit from the least significant bit (for RSA, Rabin) [1] or the most significant bit (for discrete exponentiation function) [8]. Since such a bit must be an invariant of a braid, let us consider the left-canonical form. Recall that any braid $\chi \in B_n$ is *uniquely* expressed in its left-canonical form $\chi = \Delta_n^u \chi_1 \cdots \chi_p$. Here, each of the integers u, p, and $u + p$ is called the *infimum*, the *canonical-length*, and the *supremum* of χ and denoted by $\inf(\chi)$, $\mathrm{len}(\chi)$, and $\sup(\chi)$, respectively. Because they are invariants of a braid, the hard-core bit may be derived from some of them. In contrast to (ii), the homomorphic property of the other one-way functions was essential to find their hard-core bits [8,1]. Therefore, we should approach our problem in a new way.

3.1 Candidates for the Hard-Core Bit

The following two propositions show the key properties of the infimum and the supremum to be the hard-core bits.

Proposition 1. *Let $\chi = \Delta_n^u \varphi \in B_n$, where $\varphi \in B_n^+ - \Delta_n B_n^+$. Then for any generator σ_i of B_n,*

$$\inf(\chi\sigma_i^{-1}) = \begin{cases} \inf(\chi) & \text{if } i \in F(\varphi) \\ \inf(\chi) - 1 & \text{otherwise.} \end{cases}$$

Proof. Note that for any $\chi_1, \chi_2 \in B_n$, $\inf(\chi_1\chi_2) \geq \inf(\chi_1) + \inf(\chi_2)$. Using this, we get $\inf(\chi) - 1 \leq \inf(\chi\sigma_i^{-1}) \leq \inf(\chi)$. Thus it suffices to show that $\inf(\chi\sigma_i^{-1}) = \inf(\chi)$ if and only if $i \in F(\varphi)$. If $i \in F(\varphi)$, then $\varphi = \varphi_1\sigma_i$ for some $\varphi_1 \in B_n^+ - \Delta_n B_n^+$ and $\inf(\chi\sigma_i^{-1}) = \inf(\Delta_n^u\varphi_1) = u = \inf(\chi)$. Conversely, if $\inf(\chi\sigma_i^{-1}) = \inf(\chi)$, then $\chi\sigma_i^{-1} = \Delta_n^u\varphi_2$ for some $\varphi_2 \in B_n^+ - \Delta_n B_n^+$. This implies that $\varphi = \varphi_2\sigma_i$ and so $i \in F(\varphi)$. □

Proposition 2. *Let $\Delta_n^u\chi_1 \cdots \chi_k$ be the left-canonical form of $\chi \in B_n$. Then for any generator σ_i of B_n,*

$$\sup(\chi\sigma_i) = \begin{cases} \sup(\chi) + 1 & \text{if } i \in F(\chi_k) \\ \sup(\chi) & \text{otherwise.} \end{cases}$$

Proof. If $i \in F(\chi_k)$, then it is clear that $\sup(\chi\sigma_i) = \sup(\chi) + 1$. Otherwise, $\chi_k\sigma_i$ is a permutation braid, so that $\sup(\chi\sigma_i) \leq u + k = \sup(\chi)$. Since $\sup(\chi\sigma_i) \geq \sup(\chi)$, we have $\sup(\chi\sigma_i) = \sup(\chi)$. □

From now on, we consider only the infimum. By Proposition 2, the supremum can be dealt with similarly to the infimum.

Proposition 1 shows a clue to finding a hard-core bit for the conjugacy problem in the following way: Loosely speaking, given $(\alpha, \chi^{-1}\alpha\chi)$, if an adversary is allowed to access to an oracle $\mathcal{INF}$ which on input $(\alpha, \zeta^{-1}\alpha\zeta)$ outputs $\inf(\zeta) \bmod 2$ for all $\zeta \in B_n$, then (s)he can detect the last generator of χ by comparing $\mathcal{INF}(\alpha, \chi^{-1}\alpha\chi)$ with $\mathcal{INF}(\alpha, \sigma_i\chi^{-1}\alpha\chi\sigma_i^{-1})$. In the recursive way, (s)he finally obtains the entirety of χ.

The existence of $\mathcal{INF}$ assumes that $\zeta_1^{-1}\alpha\zeta_1 = \zeta_2^{-1}\alpha\zeta_2$ implies $\inf(\zeta_1) = \inf(\zeta_2) \bmod 2$. However, it does not always happen. For example, if $\alpha = \Delta_n$ and $\zeta_2 = \Delta_n\zeta_1$, then $\zeta_1^{-1}\alpha\zeta_1 = \zeta_2^{-1}\alpha\zeta_2$ but $\inf(\zeta_2) = \inf(\zeta_1) + 1$. Since α has a major influence on the complexity of the conjugacy problem, α cannot be arbitrarily chosen but must satisfy some property.

Definition 1. *We say that $\alpha \in B_n$ is* centralizer-free *in B_m if for any $\chi \in B_m$ $(m < n)$, $\chi\alpha = \alpha\chi$ implies $\chi = e_m$.*

Note that if α is centralizer-free in B_m, then $\zeta_1^{-1}\alpha\zeta_1 = \zeta_2^{-1}\alpha\zeta_2$ $(\zeta_1, \zeta_2 \in B_m)$ implies $\zeta_1 = \zeta_2$, and hence $\inf(\zeta_1) = \inf(\zeta_2)$.

We claim that if we choose $\alpha \in B_n$ at random, then it is centralizer-free in B_{n-1} with negligible exceptions. Because the argument needs dynamics of disc homeomorphims, which seems beyond the scope of this article, we briefly list some known facts.

Fact 1. Braids are classified into three dynamical types [20,4]—periodic, reducible, pseudo-Anosov—by the Nielsen-Thurston classification of surface automorphisms [25,24,11,5]. The periodic and the reducible types are of extremely special forms and the pseudo-Anosov one is of typical form [25].

Fact 2. The pseudo-Anosov n-braids are centralizer-free in B_{n-1} (See [21]).

It seems that if we choose at random an n-braid α with p canonical factors, then it is pseudo-Anosov with probability almost $1 - \frac{1}{n^p}$.

The following proposition shows that the least significant bit of the infimum has potential for the hard-core bit for $\textsc{cnj}_\alpha$.

Proposition 3. *Let $\alpha \in B_n$ be centralizer-free in B_{n-1} and $\mathcal{INF}$ be as above. Then $\textsc{cnj}_\alpha$ is inverted for all $\chi \in B_{n-1}^+ - \Delta_{n-1}B_{n-1}^+$ by invoking $\mathcal{INF}$ polynomial in $(n, \mathrm{len}(\chi))$ times.*

Proof. We exhibit a basic algorithm that inverts $\textsc{cnj}_\alpha$ by making calls to $\mathcal{INF}$. Using Proposition 1, the algorithm on input $(\alpha, \chi^{-1}\alpha\chi)$ finds χ generator-by-generator from right to left of χ. In the middle of the execution, the variable χ' will contain the right half of the generators of χ and the variable β' is such that $\textsc{cnj}_\alpha^{-1}(\beta')$ = the left half of the χ. The algorithm, abstractly, transfers the last generator of $\textsc{cnj}_\alpha^{-1}(\beta')$ in front of χ' until $\textsc{cnj}_\alpha^{-1}(\beta') = e_{n-1}$, and thus all of χ is reconstructed in χ'.

```
1.    $\beta' \leftarrow \chi^{-1}\alpha\chi$; $\chi' \leftarrow e_{n-1}$.
2.    for $i = 1$ to $n-2$ do
2.1.      if $\mathcal{INF}(\alpha, \sigma_i\beta'\sigma_i^{-1}) = \mathcal{INF}(\alpha, \beta')$, then
              $\chi' \leftarrow \sigma_i\chi'$; $\beta' \leftarrow \sigma_i\beta'\sigma_i^{-1}$.
2.2.      if $\beta' = \alpha$, then go to step 3,
          else, go to step 2.
3.    output $\chi'$.
```

Note that every n-permutation braid is composed of at most $\frac{n(n-1)}{2}$ generators of B_n. So, the running time of the above algorithm is $\mathcal{O}(n^3\mathrm{len}(\chi)T)$, where T is the running time of $\mathcal{INF}$. □

3.2 Construction of a Collection of One-Way Functions, $\textsc{cnj}$

The original definition of one-way function refers to a single function operating on an infinite domain like $f : \{0,1\}^* \longrightarrow \{0,1\}^*$. This formulation is suitable for an abstract discussion. However, for practical purposes, an infinite collection of functions each operating on a finite domain is more adequate. In this context, this section describes a collection of one-way functions under the intractability assumption of the conjugacy problem. Recall the formal definition of a collection of one-way functions.

Definition 2 ([13]). *Let I be an index set and for each $i \in I$ let D_i be a finite domain. A* collection of one-way functions *is a set $F = \{f_i : D_i \longrightarrow \{0,1\}^*\}_{i \in I}$ satisfying the following conditions:*

Cond 1. *There exists a PPTA $\mathcal{I}$ which on input 1^n outputs $i \in I \cap \{0,1\}^n$.*
Cond 2. *There exists a PPTA $\mathcal{D}$ which on input $i \in I$ outputs $x \in D_i$.*
Cond 3. *There exists a polynomial-time algorithm that on input $(i,x) \in I \times D_i$ outputs $f_i(x)$.*
Cond 4. *For every PPTA $\mathcal{A}$, every polynomial P, and all sufficiently large n's,*

$$\Pr[f_i(z) = f_i(x) : i \leftarrow \mathcal{I}(1^n); x \leftarrow \mathcal{D}(i); z \leftarrow \mathcal{A}(i, f_i(x))] < \tfrac{1}{P(n)}.$$

Intuitively, the (n,m)-GCP becomes harder as m increases because B_m is a subgroup of B_n. As mentioned in §1, the (n,m)-GCP is a by-product of the KL-Assumption which is based on the $(n, \frac{n}{2})$-GCP [19]. However, one-way functions have no problem to be constructed from the conjugacy problem itself. To construct a hard-core predicate, from the discussion in §3.1 we consider the $(n, n{-}1)$-GCP which is almost the conjugacy problem in terms of computational complexity.

The hardness of the $(n, n-1)$-GCP depends on the braid index n, and the actual bound of the canonical-lengths of braids it takes. So it is natural and practical to take both the braid index and the canonical-length as its security parameter.

Notation. For $n \in \mathbf{N}$ and $i \le j \in \mathbf{Z}$, let $[i,j]_n \overset{\text{def}}{=} \{\chi \in B_n \mid \inf(\chi) \ge i, \sup(\chi) \le j\}$.

Construction 1. *Let $I \overset{\text{def}}{=} \{(n,p) \mid n,p \in \mathbf{N}\}$ be an index set.*

- *$\forall k = (n,p) \in I$, let $I_k \overset{\text{def}}{=} \{\alpha \in B_n^+ - \Delta_n B_n^+ \mid \operatorname{len}(\alpha) = p\}$ be an instance set. Let $\mathcal{IG}$ be a probabilistic algorithm that on input $(1^n, 1^p)$, where $k = (n,p) \in I$, outputs an element of I_k.*
- *$\forall k = (n,p) \in I$, let $D_k \overset{\text{def}}{=} [-p,p]_{n-1}$. Let $\mathcal{DG}$ be a probabilistic algorithm that on input $(1^n, 1^p)$, where $k = (n,p) \in I$, outputs an element of D_k.*
- *$\forall k = (n,p) \in I, \forall \alpha \in I_k$, define an instance function $\textsc{cnj}_\alpha : D_k \longrightarrow B_n$ by $\textsc{cnj}_\alpha(\chi) = \chi^{-1}\alpha\chi$.*
- *$\forall k = (n,p) \in I$, let F_k be the random variable defined on $\{\textsc{cnj}_\alpha\}_{\alpha \in I_k}$ distributed according to $\mathcal{IG}(1^n, 1^p)$.*
- *Let $\textsc{cnj} \overset{\text{def}}{=} \{F_k\}_{k \in I}$.*

$\textsc{cnj}$ clearly satisfies Cond 3 because given $(\alpha, \chi) \in I_{n,p} \times D_{n,p}$, one can compute the left-canonical form of $\chi^{-1}\alpha\chi$ in time $\mathcal{O}(p^2 n \log n)$ [10,19]. Now we check Cond 1,2. Notice that to satisfy Cond 4, $\mathcal{DG}(1^n, 1^p)$ cannot be mainly concentrated on polynomially many (in k) elements [13].

The proof of Theorem 3 in [19] is followed by the next corollary.

Corollary 1. *There exists a PPTA whose outputs, on input $(1^n, 1^p)$, are distributed uniformly over a subset, S, of $\{\chi \in B_n^+ - \Delta_n B_n^+ \mid \operatorname{len}(\chi) = p\}$, where $|S| \ge \left(\lfloor \frac{n-1}{2} \rfloor !\right)^p$.*

Therefore, we can have $\mathcal{IG}$ and $\mathcal{DG}$ satisfy Cond 1,2,4 under the intractability assumption of the $(n, n{-}1)$-GCP. Furthermore, from this corollary and from the discussion of α in §3.1, $\textsc{cnj}_\alpha$ can be regarded as $1-1$ for all sufficiently large $k = (n,p)$'s in I and a randomly chosen α by $\mathcal{IG}(1^n, 1^p)$. Hereafter, saying *large* k means large n and large p.

3.3 Construction of a Hard-Core Predicate, INF

This section constructs a hard-core predicate of CNJ. Recall the original definition of a hard-core predicate.

Definition 3 ([13]). *A polynomial-time computable predicate $b : \{0,1\}^* \longrightarrow \{0,1\}$ is called a* hard-core *of $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ if for every PPTA $\mathcal{A}$, every positive polynomial P, and all sufficiently large n's in* $\mathbf{N}$

$$\Pr[\mathcal{A}(f(x)) = b(x) : x \overset{u}{\leftarrow} \{0,1\}^n] < \tfrac{1}{2} + \tfrac{1}{P(n)}.$$

Notice that, given $(\alpha, \chi^{-1}\alpha\chi)$, to retrieve $\chi \in D_{n,p}$ we must know $\inf(\zeta) \bmod 2$ from $(\alpha, \zeta^{-1}\alpha\zeta)$ for many ζ's in B_{n-1} which are closely related to χ. However, any finite subset of B_{n-1} except for $\{e_{n-1}\}$ is not a group. So it happens that for some χ's in $D_{n,p}$, some ζ's are not in $D_{n,p}$. For this reason, the domain of hard-core predicate is defined slightly different from the corresponding one of CNJ.

For every $k = (n,p) \in I$, consider a slightly enlarged set of D_k,

$$\bar{D}_k \overset{\text{def}}{=} D_k \cup \{\chi\sigma_i^{-1} \mid \chi \in D_k, i \in \{1, \dots, n-2\}\}.$$

Thus, $D_k = [-p,p]_{n-1} \subset \bar{D}_k \subset [-(p+1),p]_{n-1} \subset D_{n,p+1}$.

Notation. $\sigma_0 \overset{\text{def}}{=} e_n$.

For every $k = (n,p) \in I$, define a PPTA $\overline{\mathcal{DG}}(1^n, 1^p)$ in the following order:

$$\chi \leftarrow \mathcal{DG}(1^n, 1^p);\ i \overset{u}{\leftarrow} \{0, 1, \dots, n-2\};\ \text{output } \chi\sigma_i^{-1}.$$

Using the infimum and $\bar{D}_k$, we now define a collection of boolean predicates

$$\text{INF} = \{\text{INF}_k : \bar{D}_k \longrightarrow \{0,1\}\}_{k \in I} \ \text{ by } \ \text{INF}_k(\chi) = \inf(\chi) \bmod 2.$$

The following lemma is crucial to our main result. It shows, for a random choice $\chi \in \bar{D}_k$, how to turn a PPTA that predicts correctly $\text{INF}_k(\chi)$ from $\text{CNJ}_\alpha(\chi)$ with probability non-negligibly higher than 1/2 into a PPTA predicting almost correctly.

Lemma 1. *For an infinite subset F of I, let $\mathcal{A}$ be a PPTA and P be a positive polynomial such that for all $k = (n,p) \in F$*

$$\Pr[\mathcal{A}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) = \text{INF}_k(\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)] \geq \tfrac{1}{2} + \tfrac{1}{P(k)}.$$

Then for any positive polynomial Q, there exists a PPTA $\mathcal{C}$ such that for all $k = (n,p) \in F$

$$\Pr[\mathcal{C}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) = \text{INF}_k(\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)] \geq 1 - \tfrac{1}{Q(k)}.$$

Proof. For every $k \in F$, let $N = N(k) \overset{\text{def}}{=} \frac{1}{4}P(k)^2Q(k)$. On every input $(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)$, where $k = (n,p) \in F, \alpha \in [\mathcal{IG}(1^n, 1^p)]$, and $\chi \in [\overline{\mathcal{DG}}(1^n, 1^p)]$, $\mathcal{C}$ executes the following algorithm:

1. Invoke $\mathcal{A}$ on input $(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)$ independently N-times. And let $\mathcal{A}^{(i)}$ be the i-th invoking of $\mathcal{A}$ for each $i \in \{1, \dots, N\}$.
2. If $\sum_{i=1}^{N} \mathcal{A}^{(i)}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) \geq \frac{N}{2}$, output 1. Otherwise, output 0.

For every $k = (n,p) \in I$ and every $i \in \{1, \dots, N\}$, define a PPTA $\zeta_i^{\mathcal{A}}(1^n, 1^p, \cdot, \cdot)$ induced by $\mathcal{A}$ as

$$\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) = \begin{cases} 1 & \text{if } \mathcal{A}^{(i)}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) \neq \text{INF}_k(\chi) \\ 0 & \text{otherwise,} \end{cases}$$

where $\alpha \leftarrow \mathcal{IG}(1^n, 1^p)$; $\chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)$.

The independence of $\langle \{\mathcal{A}^{(i)}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)\}_{1 \leq i \leq N} : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\rangle$ yields the independence of $\langle \{\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)\}_{1 \leq i \leq N} : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\rangle$. And for every $i \in \{1, \dots, N\}$

$$\begin{aligned}
&\Pr\left[\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\right] \\
&\quad = \Pr\left[\mathcal{A}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) \neq \text{INF}_k(\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\right] \\
&\quad \leq \tfrac{1}{2} - \tfrac{1}{P(k)}.
\end{aligned}$$

So $\langle \{\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)\}_{1 \leq i \leq N} : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\rangle$ are independent and identically distributed random variables with common binomial distribution $B(1,p)$, where $p \leq \frac{1}{2} - \frac{1}{P(k)}$.

From $\mathrm{E}[\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)] \leq \frac{1}{2} - \frac{1}{P(k)}$ and by applying Chebyshev's inequality, we get

$$\begin{aligned}
&\Pr\left[\frac{1}{N}\sum_{i=1}^{N} \zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) \geq \frac{1}{2} : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\right] \\
&\quad \leq P(k)^2 \mathrm{Var}\left[\frac{1}{N}\sum_{i=1}^{N} \zeta_i^{\mathcal{A}}(1^n,1^p, \alpha, \chi^{-1}\alpha\chi) : \alpha \leftarrow \mathcal{IG}(1^n,1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n,1^p)\right].
\end{aligned}$$

Because $\langle \{\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)\}_{1 \leq i \leq N} : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\rangle$ are pairwise independent and because

$$\mathrm{Var}[\zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)] < \tfrac{1}{4},$$

it follows that

$$\mathrm{Var}\left[\frac{1}{N}\sum_{i=1}^{N} \zeta_i^{\mathcal{A}}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)\right] < \tfrac{1}{4N}.$$

Thus,

$$\Pr\left[\tfrac{1}{N}\textstyle\sum_{i=1}^{N} \zeta_i^{\mathcal{A}}(1^n,1^p, \alpha, \chi^{-1}\alpha\chi) \geq \frac{1}{2} : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n,1^p)\right] < \tfrac{1}{Q(k)}.$$

That is to say,

$$\Pr[\mathcal{C}(1^n,1^p, \alpha, \chi^{-1}\alpha\chi) = \text{INF}_k(\chi) : \alpha \leftarrow \mathcal{IG}(1^n,1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n,1^p)] \geq 1 - \tfrac{1}{Q(k)}.$$

$\square$

By this lemma and by the basic algorithm in Proposition 3, we get the following result:

Theorem 1. INF *is a hard-core predicate of* CNJ.

Proof. Assume that there exist a PPTA $\mathcal{A}$, an infinite subset F of I, and a positive polynomial P such that for all $k = (n, p) \in F$

$$\Pr[\mathcal{A}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) = \textsc{inf}_k(\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)] \geq \tfrac{1}{2} + \tfrac{1}{P(k)}.$$

From Lemma 1, there is a PPTA $\mathcal{C}$ such that for all $k = (n, p) \in F$

$$\Pr[\mathcal{C}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi) = \textsc{inf}_k(\chi) : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \overline{\mathcal{DG}}(1^n, 1^p)] \geq 1 - \tfrac{1}{2pn^3}.$$

Fix $k = (n, p) \in F$. Using the basic algorithm in Proposition 3, on input $(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)$, where $\alpha \leftarrow \mathcal{IG}(1^n, 1^p)$; $\chi \leftarrow \mathcal{DG}(1^n, 1^p)$, $\mathcal{M}$ executes the following algorithm:

```
1.      $\beta' \leftarrow \chi^{-1}\alpha\chi$; $\chi' \leftarrow e_{n-1}$.
2.      for $u = -p$ to $p$ do
2.1.        if $\beta' = \Delta_{n-1}^{-u}\alpha\Delta_{n-1}^{u}$, then go to step 4.
3.      for $j = 1$ to $n-1$ do
3.1.        $i \overset{u}{\leftarrow} \{0, 1, \ldots, n-2\}$.
3.2.        if $\mathcal{C}(1^n, 1^p, \alpha, \sigma_i\beta'\sigma_i^{-1}) = \mathcal{C}(1^n, 1^p, \alpha, \beta')$, then
                $\chi' \leftarrow \sigma_i\chi'$; $\beta' \leftarrow \sigma_i\beta'\sigma_i^{-1}$,
            else go to step 3.
3.3.        for $u = -p$ to $p$ do
3.3.1.          if $\beta' = \Delta_{n-1}^{-u}\alpha\Delta_{n-1}^{u}$, then go to step 4.
3.4.        go to step 3.
4.      output $\Delta_{n-1}^{u}\chi'$.
```

Each repetition of the above algorithm makes two calls to $\mathcal{C}$ independently and the number of repetitions of the algorithm is at most $p(n-1)^3$. By the definition of $\overline{\mathcal{DG}}$ and $\mathcal{M}$, for all $k = (n, p) \in F$

$$\Pr\left[\zeta^{-1}\alpha\zeta = \chi^{-1}\alpha\chi : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \leftarrow \mathcal{DG}(1^n, 1^p); \zeta \leftarrow \mathcal{M}(1^n, 1^p, \alpha, \chi^{-1}\alpha\chi)\right] > \tfrac{1}{2pn^3}.$$

□

Notice that hard-core predicates are used to construct pseudorandom generators in some cases by Blum-Micali's general method [8]. Loosely speaking, if $l : \mathbf{N} \longrightarrow \mathbf{N}$ is a stretching function and $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ is a $1-1$ one-way function with a hard-core b, then $G(s) \overset{\text{def}}{=} b(x_1)b(x_2)\cdots b(x_{l(n)})$ is a pseudorandom generator, where $x_0 = s$ and $x_i = f(x_{i-1})$ for $i = 1, \ldots, l(n)$. This method does not apply to INF because $\textsc{cnj}_\alpha(D_k)$ is much larger than D_k. From the fact that most known one-way functions in braid groups (see [19]) do not preserve their finite domains, hard-core predicates in braid groups seem to have no relation to this method.

4 Pseudorandom Schemes

The original KL-Assumption is as follows:

> Given a triplet $(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi)$ of elements in B_n, where $\chi \in \langle\sigma_1, \ldots, \sigma_{\lfloor\frac{n}{2}\rfloor-1}\rangle$ and $\psi \in \langle\sigma_{\lfloor\frac{n}{2}\rfloor+1}, \ldots, \sigma_{n-1}\rangle$, it is computationally infeasible to find $\psi^{-1}\chi^{-1}\alpha\chi\psi$.

Let $m(n) \overset{\text{def}}{=} \lfloor\frac{n}{2}\rfloor$. For every $k = (n,p) \in I$, let m mean $m(n)$ and let $LD_k \overset{\text{def}}{=} [-p,p]_m$. Consider a group monomorphism $\tau : B_{n-m} \longrightarrow B_n$ defined by $\tau(\sigma_i) = \sigma_{m+i}$ for $i = 1, \ldots, n-m-1$. Then $\tau(B_{n-m}) = \langle\sigma_{m+1}, \ldots, \sigma_{n-1}\rangle$ is a subgroup of B_n isomorphic to B_{n-m}. Let $RD_k \overset{\text{def}}{=} \tau([-p,p]_{n-m})$. Here, we defined $m(n)$ as $\lfloor\frac{n}{2}\rfloor$ for notational convenience. Instead, it can take any number around this. From the definition of LD_k and RD_k, for every $k = (n,p) \in I$ and every $(\chi, \psi) \in LD_k \times RD_k$, it follows that: (i) $\chi\psi = \psi\chi$, (ii) $\chi\psi \in [-p,p]_n$. (i) is trivial. (ii) uses the fact that there exists $\zeta \in B_n^+$ such that $\Delta_n = \Delta_m\tau(\Delta_{n-m})\zeta$.

For every $k = (n,p) \in I$ and every $\alpha \in I_k$, let $R_{k,\alpha} \overset{\text{def}}{=} \{\zeta^{-1}\alpha\zeta \mid \zeta \in [-p,p]_n\}$. Using these notations, the DKL-Assumption is stated as follows:

[The DKL-Assumption]

For every PPTA $\mathcal{A}$, every positive polynomial P, and all sufficiently large $k = (n,p)$'s in I,

$$\Big|\Pr\big[\mathcal{A}(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \psi^{-1}\chi^{-1}\alpha\chi\psi) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \overset{u}{\leftarrow} LD_k; \psi \overset{u}{\leftarrow} RD_k\big]$$
$$- \Pr\big[\mathcal{A}(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \beta) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \overset{u}{\leftarrow} LD_k; \psi \overset{u}{\leftarrow} RD_k; \beta \overset{u}{\leftarrow} R_{k,\alpha}\big]\Big|$$
$$< \frac{1}{P(k)}.$$

Actually, there is no known PPTA sampling χ from LD_k uniformly at random. However, from Corollary 1, one can construct a PPTA $\mathcal{LDG}$ such that for every $k = (n,p) \in I$, $\mathcal{LDG}(1^n, 1^p)$ is uniformly distributed on $[\mathcal{LDG}(1^n, 1^p)] \subset LD_k$. Moreover, for every polynomial Q, $|[\mathcal{LDG}(1^n, 1^p)]| > Q(k)$ for all sufficiently large $k = (n,p)$'s in I. So, in this section saying that $\chi \overset{u}{\leftarrow} LD_k$ implicitly means two folds. On the one hand, we have such a $\mathcal{LDG}$ as this. On the other hand, $\chi \leftarrow \mathcal{LDG}(1^n, 1^p)$. In other words, LD_k means $[\mathcal{LDG}(1^n, 1^p)]$ in a probabilistic sense. Likewise, let us view $\chi \overset{u}{\leftarrow} RD_k$ and $\chi \overset{u}{\leftarrow} R_{k,\alpha}$ in this way.

Under this DKL-Assumption, this section constructs a pseudorandom generator and a pseudorandom synthesizer which are similar to those based on the decision Diffie-Hellman assumption [22]. Since the securities are proved typically by the standard hybrid techniques [13,16,22,23], we only sketch them.

4.1 Pseudorandom Generator

Recall the formal definition of pseudorandom generator.

Definition 4 ([26,8]). *A deterministic polynomial-time algorithm, $G : \{0,1\}^* \longrightarrow \{0,1\}^*$, is called a* pseudorandom generator *if there exists a stretching function, $l : \mathbf{N} \longrightarrow \mathbf{N}$, so that for all $x \in \{0,1\}^*$, $||G(x)|| = l(||x||)$ and if for every PPTA $\mathcal{A}$, every positive polynomial P, and all sufficiently large n's in $\mathbf{N}$*

$$\left|\Pr[\mathcal{A}(G(x)) = 1 : x \overset{u}{\leftarrow} \{0,1\}^n] - \Pr[\mathcal{A}(r) = 1 : r \overset{u}{\leftarrow} \{0,1\}^{l(n)}]\right| < \tfrac{1}{P(n)}.$$

The idea of this section is as follows: Given $(\alpha, \chi^{-1}\alpha\chi)$ for $\alpha \in B_n, \chi \in B_m$, it looks hard to find χ even if we know $(\psi_i^{-1}\alpha\psi_i, \chi^{-1}\psi_i^{-1}\alpha\psi_i\chi)$'s for polynomially many ψ_i's randomly chosen in $\tau(B_{n-m})$.

Notation. For every $k \in I$ and every $\alpha \in I_k$, let $LR_{k,\alpha} \overset{\text{def}}{=} \{\chi^{-1}\alpha\chi \mid \chi \in LD_k\}$.

Definition 5 ($\mathcal{PGIG}_{KL}$). *An* instance generator *$\mathcal{PGIG}_{KL}$ is a probabilistic algorithm that on input $(1^n, 1^p, 1^l)$, where $k = (n,p) \in I$ and $l \in \mathbf{N}$, executes the following:*

$$\alpha \leftarrow \mathcal{IG}(1^n, 1^p);\ \alpha_1, \dots, \alpha_l \overset{u}{\leftarrow} LR_{k,\alpha};\ \textit{output } (\alpha, \alpha_1, \dots, \alpha_l).$$

By the definition of $\mathcal{IG}$ in §3.2, $\mathcal{PGIG}_{KL}$ clearly runs in polynomial in (k,l) time.

Construction 2. *Let $l : I \longrightarrow \mathbf{N}$ be a polynomial. For every $k = (n,p) \in I$, $\alpha \in I_k$, $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_l) \in (LR_{k,\alpha})^l$, define $g_{\alpha,\boldsymbol{\alpha}} : RD_k \longrightarrow (R_{k,\alpha})^l$ by $g_{\alpha,\boldsymbol{\alpha}}(\psi) = (\psi^{-1}\alpha_1\psi, \dots, \psi^{-1}\alpha_l\psi)$, where $l = l(k)$. Let G_k be the random variable that assumes as values the function $g_{\alpha,\boldsymbol{\alpha}}$, where the distribution of $(\alpha, \boldsymbol{\alpha})$ is $\mathcal{PGIG}_{KL}(1^n, 1^p, 1^l)$. Let $G_{KL} \overset{\text{def}}{=} \{G_k\}_{k \in I}$.*

The following result shows that G_{KL} is pseudorandom at least as secure as the DKL-Assumption.

Theorem 2. *If the DKL-Assumption holds, then for every PPTA $\mathcal{A}$, every positive polynomial P, and all sufficiently large $k = (n,p)$'s in I,*

$$\begin{aligned}&\Big|\Pr\big[\mathcal{A}(g_{\alpha,\boldsymbol{\alpha}}(\psi)) = 1 : (\alpha, \boldsymbol{\alpha}) \leftarrow \mathcal{PGIG}_{KL}(1^n, 1^p, 1^l); \psi \overset{u}{\leftarrow} RD_k\big]\\ &\quad - \Pr\big[\mathcal{A}(\beta_1, \dots, \beta_l) = 1 : \alpha \leftarrow \mathcal{IG}(k); \beta_1, \dots, \beta_l \overset{u}{\leftarrow} R_{k,\alpha}\big]\Big|\\ &\quad < \tfrac{1}{P(k)},\end{aligned}$$

where $l = l(k)$.

Sketch of Proof. Fix $k = (n,p) \in I$ and let $l = l(k)$. First, define a PPTA $\mathcal{M}$, on input $\langle \alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \tilde{\beta}\rangle$ where $\alpha \in I_k$, $\chi \in LD_k$, $\psi \in RD_k$, and $\tilde{\beta} \in R_{k,\alpha}$, from $\mathcal{A}$ as:

1. $J \overset{u}{\leftarrow} \{1, \dots, l\}$.
2. $\chi_1, \dots, \chi_{J-1} \overset{u}{\leftarrow} LD_k;\ \beta_{J+1}, \dots, \beta_l \overset{u}{\leftarrow} R_{k,\alpha}$.
3. $H \overset{\text{def}}{=} \langle \chi_1^{-1}\psi^{-1}\alpha\psi\chi_1, \dots, \chi_{J-1}^{-1}\psi^{-1}\alpha\psi\chi_{J-1}, \tilde{\beta}, \beta_{J+1}, \dots, \beta_l\rangle$.

4. Output $\mathcal{A}(H)$.

Next, for each $i \in \{1, \dots, l\}$, define the i-th hybrid distribution

$$H_i^{k,l} = \langle \psi^{-1}\alpha_1\psi, \dots, \psi^{-1}\alpha_i\psi, \beta_{i+1}, \dots, \beta_l \rangle,$$

where $(\alpha, \alpha_1, \dots, \alpha_i) \leftarrow \mathcal{PGIG}_{KL}(1^n, 1^p, 1^i)$; $\psi \overset{u}{\leftarrow} RD_k$; $\beta_{i+1}, \dots, \beta_l \overset{u}{\leftarrow} R_{k,\alpha}$.

Then we get that

$$\begin{aligned}
&\Big|\Pr\big[\mathcal{M}(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \psi^{-1}\chi^{-1}\alpha\chi\psi) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \overset{u}{\leftarrow} LD_k; \psi \overset{u}{\leftarrow} RD_k\big] \\
&\quad - \Pr\big[\mathcal{M}(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \beta) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \overset{u}{\leftarrow} LD_k; \psi \overset{u}{\leftarrow} RD_k; \beta \overset{u}{\leftarrow} R_{k,\alpha}\big]\Big| \\
&= \frac{1}{l}\Big|\Pr\big[\mathcal{A}(H_l^{k,l}) = 1 : (\alpha, \alpha_1, \dots, \alpha_l) \leftarrow \mathcal{PGIG}_{KL}(1^n, 1^p, 1^l); \psi \overset{u}{\leftarrow} RD_k\big] \\
&\quad - \Pr\big[\mathcal{A}(H_0^{k,l}) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \beta_1, \dots, \beta_l \overset{u}{\leftarrow} R_{k,\alpha}\big]\Big|.
\end{aligned}$$

Using these, the theorem can be proved by contradiction. □

So, G_{KL} generates pseudorandom sequences of braids in $R_{k,\alpha}$. A pseudorandom generator can be constructed from G_{KL} by making use of the leftover hash lemma and pairwise independent hash functions [18,16,22].

The expansion property of the pseudorandom generator depends on the choice of $l(\cdot)$. Namely, $l(\cdot)$ should satisfy: $l(k)\log_2|R_{k,\alpha}| > 2||RD_k||$. Using the fact that $|R_{k,\alpha}| \geq |LD_k| \cdot |RD_k|$, $l(n,p) = 2pn$ suffices.

4.2 Pseudorandom Synthesizer

Although the notion of pseudorandom synthesizer was first introduced by Naor *et al.* [23] as a useful tool to get a parallel construction of a pseudorandom function, it is important itself as another type of pseudorandom generator. More precisely, pseudorandom synthesizers may be useful for software implementations of pseudorandom generators because from a pseudorandom synthesizer a pseudorandom generator with long output length can be easily defined and subsequences of its output can be computed directly.

Recall the formal definition of a pseudorandom synthesizer:

Notation ([23]). Let $f : \{0,1\}^{2n} \longrightarrow \{0,1\}^l$ be any function, and let $x = (x_1, \dots, x_k)$ and $y = (y_1, \dots, y_m)$ be two sequences of n-bit strings. We define $\mathbf{C}_f(x,y)$ to be the $(k \times m)$-matrix $(f(x_i, y_j))_{i,j}$.

Definition 6 ([23]). *Let $l : \mathbf{N} \longrightarrow \mathbf{N}$ be any function, and let $S : \{0,1\}^* \times \{0,1\}^* \longrightarrow \{0,1\}^*$ be a polynomial-time computable function such that for every $x, y \in \{0,1\}^n$, $||S(x,y)|| = l(n)$. Then S is a* pseudorandom synthesizer *if for every PPTA $\mathcal{A}$, every two positive polynomials P and m, and all sufficiently large n's*

$$|\Pr[\mathcal{A}(\mathbf{C}_S(x,y)) = 1] - \Pr[\mathcal{A}((r_{i,j})_{1\leq i,j\leq m}) = 1]| < \tfrac{1}{P(n)},$$

where $m = m(n)$ and $x_1, \dots, x_m, y_1, \dots, y_m \overset{u}{\leftarrow} \{0,1\}^n$; $x = (x_1, \dots, x_m)$, $y = (y_1, \dots, y_m)$; $r_{1,1}, \dots, r_{m,m} \overset{u}{\leftarrow} \{0,1\}^{l(n)}$.

As mentioned in §1.1, the notion of pseudorandom synthesizer is stronger because pseudorandom synthesizers require that $\{S(z_i)\}_{1\le i\le m^2}$ remains pseudorandom even when the z_i's are of the form $\{x_i \circ y_j\}_{1\le i,j\le m}$, where $\circ$ stands for x concatenated with y. If $l(n) > 2n$ for all $n \in \mathbf{N}$, a pseudorandom synthesizer directly becomes a pseudorandom generator with $m(n) = 1$. However, every pseudorandom generator is not a pseudorandom synthesizer (See [23] for example).

Now we construct a pseudorandom synthesizer based on the DKL-Assumption.

Construction 3. *For every $k = (n, p) \in I$ and every $\alpha \in I_k$, define $s_\alpha : LD_k \times RD_k \longrightarrow R_{k,\alpha}$ by $s_\alpha(\chi, \psi) = \psi^{-1}\chi^{-1}\alpha\chi\psi$. Let S_k be the random variable that assumes as values the function s_α according to the distribution, $\mathcal{IG}(1^n, 1^p)$. Let $S_{KL} \overset{\text{def}}{=} \{S_k\}_{k\in I}$.*

Then we get the following result:

Theorem 3. *If the DKL-Assumption holds, then for every PPTA $\mathcal{A}$, every positive polynomials l, P, and all sufficiently large $k = (n, p)$'s in I,*

$$|\Pr[\mathcal{A}(\mathbf{C}_{s_\alpha}(\chi, \psi)) = 1] - \Pr[\mathcal{A}((\gamma_{i,j})_{1\le i,j\le l}) = 1]| < \tfrac{1}{P(k)},$$

where $l = l(k)$ and $\alpha \leftarrow \mathcal{IG}(1^n, 1^p)$; $\chi_1, \dots, \chi_l \overset{u}{\leftarrow} LD_k$; $\chi = (\chi_1, \dots, \chi_l)$; $\psi_1, \dots, \psi_l \overset{u}{\leftarrow} RD_k$; $\psi = (\psi_1, \dots, \psi_l)$; $\gamma_{1,1}, \dots, \gamma_{l,l} \overset{u}{\leftarrow} R_{k,\alpha}$.

Sketch of Proof. Fix $k = (n, p) \in I$ and let $l = l(k)$. First, define a PPTA $\mathcal{M}$, on input $\langle \alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \tilde{\beta}\rangle$ where $\alpha \in I_k$, $\chi \in LD_k$, $\psi \in RD_k$, and $\tilde{\beta} \in R_{k,\alpha}$, from $\mathcal{A}$ as:

1. $J \overset{u}{\leftarrow} \{1, \dots, l^2\}$.
2. Compute J_1, J_2 such that $1 \le J_1, J_2 \le l$ and $J = l(J_1 - 1) + J_2$.
3. Let $\chi_{J_1} \overset{\text{def}}{=} \chi$ and $\psi_{J_2} \overset{\text{def}}{=} \psi$.
4. $\chi_1, \dots, \chi_{J_1-1} \overset{u}{\leftarrow} LD_k$; $\psi_1, \dots, \psi_{J_2-1}, \psi_{J_2+1}, \dots, \psi_l \overset{u}{\leftarrow} RD_k$; $\beta_{J+1}, \dots, \beta_{l^2} \overset{u}{\leftarrow} R_{k,\alpha}$.
5. Define the $(l \times l)$-matrix $H = (h_{i,j})_{1\le i,j\le l}$ to be
$$h_{i,j} = \begin{cases} \psi_j^{-1}\chi_i^{-1}\alpha\chi_i\psi_j & \text{if } l(i-1)+j < J, \\ \tilde{\beta} & \text{if } l(i-1)+j = J, \\ \beta_w & \text{if } w \overset{\text{def}}{=} l(i-1)+j > J. \end{cases}$$
6. Output $\mathcal{A}(H)$.

Next, for each $0 \le r \le l^2$, define the r-th hybrid distribution $H_r^{k,l} = (h_{i,j})_{1\le i,j\le l}$ to be

$$h_{i,j} = \begin{cases} \psi_j^{-1}\chi_i^{-1}\alpha\chi_i\psi_j & \text{if } l(i-1)+j \le r, \\ \beta_w & \text{if } w \overset{\text{def}}{=} l(i-1)+j > r, \end{cases}$$

where $\alpha \leftarrow \mathcal{IG}(1^n, 1^p)$; $\chi_1, \ldots, \chi_l \overset{u}{\leftarrow} LD_k$; $\psi_1, \ldots, \psi_l \overset{u}{\leftarrow} RD_k$; $\beta_{r+1}, \ldots, \beta_{l^2} \overset{u}{\leftarrow} R_{k,\alpha}$.

Then we get that

$$\Big|\Pr\big[\mathcal{M}(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \psi^{-1}\chi^{-1}\alpha\chi\psi) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \overset{u}{\leftarrow} LD_k; \psi \overset{u}{\leftarrow} RD_k\big]$$
$$- \Pr\big[\mathcal{M}(\alpha, \chi^{-1}\alpha\chi, \psi^{-1}\alpha\psi, \beta) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi \overset{u}{\leftarrow} LD_k; \psi \overset{u}{\leftarrow} RD_k; \beta \overset{u}{\leftarrow} R_{k,\alpha}\big]\Big|$$
$$= \frac{1}{l^2}\Big|\Pr\big[\mathcal{A}(H_{l^2}^{k,l}) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \chi_1, \ldots, \chi_l \overset{u}{\leftarrow} LD_k; \psi_1, \ldots \psi_l \overset{u}{\leftarrow} RD_k\big]$$
$$- \Pr\big[\mathcal{A}(H_0^{k,l}) = 1 : \alpha \leftarrow \mathcal{IG}(1^n, 1^p); \beta_1, \ldots, \beta_{l^2} \overset{u}{\leftarrow} R_{k,\alpha}\big]\Big|.$$

Using these, the theorem can be proved by contradiction. □

5 Concluding Remarks

This article has considered two related hard problems in braid groups: the conjugacy and the Ko-Lee problems, which are believed to be computationally infeasible in our current state of knowledge.

Assuming that the conjugacy problem is one-way, we have presented two peculiar hard-core predicates that are provably secure using the infimum and the supremum of a braid. This means that, given $(\alpha, \chi^{-1}\alpha\chi)$, predicting the least significant bit of $\inf(\chi)$ $(\mathit{or}\ \sup(\chi))$ is as hard as the entirety of χ.

Under the decision Ko-Lee assumption, we have proposed two practical pseudorandom schemes, a pseudorandom generator and a pseudorandom synthesizer, that are provably secure.

Braid groups are quite different from the other groups which have been dealt with so far. So the known methods to turn hard-core predicates into pseudorandom generators and to turn pseudorandom generators or pseudorandom synthesizers into pseudorandom function generators cannot be applied naively. Therefore, a natural line for further research is to study how to get these next cryptographic primitives from our results.

Acknowledgement. We wish to thank the members of the CRYPTO committee. The first two authors were supported in part by the Ministry of Science and Technology under the National Research Laboratory Grant 1999 program. And the third author was supported in part by the Korea IT Industry Promotion Agency under the Information Technologies Research Center program.

References

1. W. Alexi, B. Chor, O. Goldreich, and C.P. Schnorr, *RSA and Rabin functions: certain parts are as hard as the whole*, SIAM J. Comput. **17** (1988) 194–209.
2. I. Anshel, M. Anshel, and D. Goldfeld, *An algebraic method for public-key cryptography*, Math. Res. Lett. **6** (1999) 287–291.
3. M. Bellare and P. Rogaway, *Random oracles are Practical: a Paradigm for Designing Efficient Protocols*, In 1st Annual Conference on Computer and Communications Security, ACM (1993) 62–73.

4. D. Bernardete, Z. Nitecki and M. Gutierrez, *Braids and the Nielsen-Thurston classification*, J. Knot theory and its ramifications **4** (1995) 549–618.
5. M. Bestvina and M. Handel, *Train Tracks for surface automorphisms*, Topology **34** (1995) 109–140.
6. J. Birman, *Braids, links and the mapping class group*, Ann. Math. Studies 82, Princeton Univ. Press (1974).
7. J.S. Birman, K.H. Ko, and S.J. Lee, *New approaches to the world and conjugacy problem in the braid groups*, Advances in Math. **139** (1998) 322–353.
8. M. Blum and S. Micali, *How to generate cryptographically strong sequences of pseudorandom bits*, SIAM J. Comput. **13** (1984) 850–864.
9. W. Diffie and M.E. Hellman, *New Directions in Cryptography*, IEEE Trans. on Info. Theory, IT-22 (1976) 644-654.
10. D.B.A. Epstein, J.W. Cannon, D.F. Holt, S.V.F. Levy, M.S. Patterson, and W. Thurston, *Word processing in groups*, Jones and Barlett, Boston and London (1992).
11. A. Fathi, F. Laudenbach, and V. Poénaru, *Travaux de Thurston sur les surfaces*, Astérisque (1979) 66–67.
12. J.B. Fischer and J. Stern, *An Efficient Pseudo-Random Generator Provably as Secure as Syndrome Decoding*, Proc. Eurocrypt '96, LNCS 1070, Springer-Verlag (1996) 245–255.
13. O. Goldreich, *Foundation of Cryptography—Fragments of a Book*, Available at `http://www.theory.lcs.mit.edu/~oded/frag.html` (1995).
14. O. Goldreich and L.A. Levin, *Hard-core Predicates for any One-Way Function*, 21st STOC (1989) 25–32.
15. S. Goldwasser, S. Micali, and R. Rivest, *A Digital signature scheme secure against adaptive chosen-message attacks*, SIAM J. Comput. **17** (1988) 281–308.
16. J. Hastad, R. Impaglizo, L.A. Levin, and M. Luby, *A Pseudorandom Generator from any One-way Function*, SIAM J. Comput. **28** (1999) 1364–1396.
17. R. Impagliazzo and M. Naor, *Efficient cryptographic schemes provably as secure as subset sum*, Proc. IEEE 30th Symp. on Found. of Comput. Sci. (1989) 231–241.
18. R. Impagliazzo and D. Zuckerman, *Reclying random bits*, Proc. 30th IEEE Symposium on Foundations of Computer Science (1989) 248–253.
19. K.H. Ko, S.J. Lee, J.H. Cheon, J.W. Han, J.S. Kang, and C. Park, *New Public-key Cryptosystem Using Braid Groups*, Proc. Crypto 2000, LNCS 1880, Springer-Verlag (2000) 166–183.
20. J. Los, *Pseudo-Anosov maps and invariant train track in the disc: a finite algorithm*, Proc. Lond. Math. Soc. **66** (1993) 400-430.
21. J. McCarthy, *Normalizers and centralizers of psuedo-Anosov mapping clases*, Available at `http://www.mth.msu.edu/~mccarthy/research/`.
22. M. Naor and O. Reingold, *Number-Theoretic constructions of efficient pseudo-random functions*, Proc. 38th IEEE Symp. on Foundations of Computer Science (1997) 458–467.
23. M. Naor and O. Reingold, *Synthesizers and their application to the parallel construction of pseudo-random functions*, J. of Computer and Systems Sciences **58** (1999) 336–375.
24. J. Nielsen, In *Collected papers of J. Nielsen*, Birkhauser (1986).
25. W. P. Thurston, *On the geometry and dynamics of diffeomorphisms of surfaces*, Bull. AMS **19** (1988) 417-431.
26. A. Yao, *Theory and applications of trapdoor functions*, Proc. 23rd IEEE Symp. of Found. of Comput. Sci. (1982) 80–91.