Math 109: Groups and Symmetry

Department of Mathematics, Stanford University
Fall Quarter, 2021

Braid Groups and the Conjugacy Search Problem in Cryptography

Writing in the Major

Matthew Ding

December 7, 2021

Contents

1 Introduction to Cryptography

The study of cryptography is motivated by the problem of communicating securely over an insecure channel. Naturally, solutions to this problem have taken the form of complex mathematical encryptions that make deciphering messages computationally difficult if not impossible. And with the development of more powerful computers over the last century, our cryptographic methods have similarly improved. Modern cryptographic algorithms (also called schemes or protocols) are now generally based on hard problems in computation theory. Solutions to these problems are presumed to be so inefficient that it would be uneconomical for an adversary to solve them. Of course, this assumption also means that if efficient solutions to these protocols were ever found, the protocols would be rendered insecure and susceptible to attack.

The standard model for a cryptographic scheme involves two parties, Alice and Bob, each of whom possess a shared secret key. Using this key, they are able to easily encrypt and decrypt any messages that they send between each other while keeping their messages indecipherable to eavesdroppers.

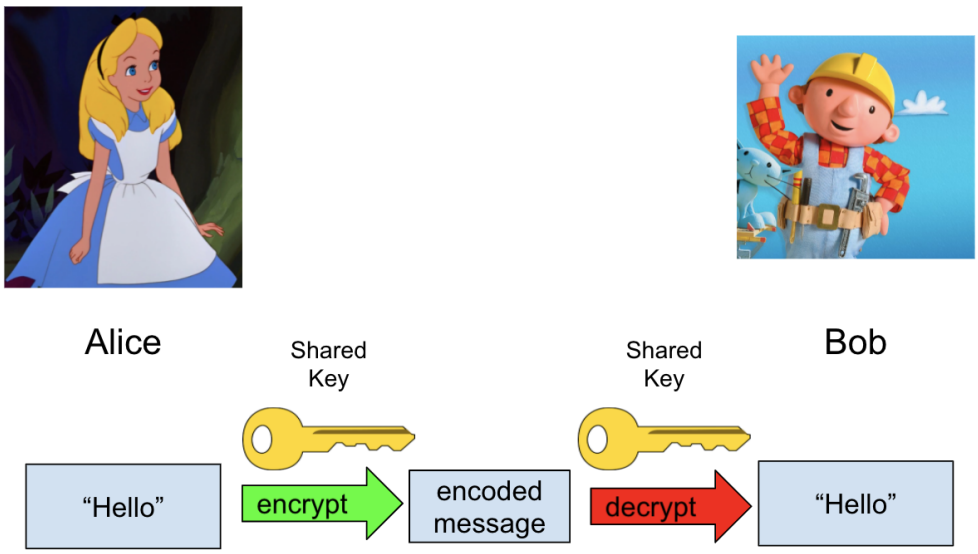

Figure 1: A Diagram of a Cryptographic Model

Such a model where Alice and Bob share the same secret key is called a *symmetric key cypher*. Traditionally, secure communication required two parties to exchange keys beforehand by some physical means. However, the development of public-key cryptosystems, largely based off of group theoretic concepts, allows for two parties without a secret key to generate one. They do so by employing a *public key* that is available to everyone. If done right, such an system preserves privacy while allowing any two parties to easily generate a secret key.

This paper specifically focuses on public-key cryptosystems. We start in section one by presenting the Diffie-Hellman Protocol, which was the first to introduce the concept. The rest of the paper centers around public-key cryptosystems based on the *braid group*. We formally define this group in section two before introducing the Ko et al. Protocol, a braid group variation of Diffie-Hellman. The final part of the paper discusses the open Conjugacy Search Problem in group theory, an efficient solution to which renders the Ko et al. protocol insecure.

1.1 Diffie-Hellman Protocol

The version of the Diffie-Hellman protocol I introduce in this paper uses cyclic groups to generate a key. It is largely based on the following elementary group property:

Proposition 1. *Let g be any element of a group. Then for $m, n \in \mathbb{Z}$, we have $(g^m)^n = (g^n)^m$.*

Proof. Because properties of exponentiation extend to groups, we have $(g^m)^n = g^{mn} = g^{nm} = (g^n)^m$. □

Now, the protocol is laid out as follows [4]:

Diffie-Hellman Key Exchange: *Let G be a cyclic group and g be a generator of G where both g and its order d are publicly known. If Alice and Bob wish to create a shared key, they execute the following protocol:*

1. *Alice selects uniformly at random an integer $a \in [2, d-1]$ and stores it. She then computes g^a and sends it to Bob.*
2. *Bob selects uniformly at random an integer $b \in [2, d-1]$ and stores it. He then computes g^b and sends it to Alice.*
3. *Alice computes $k_a = (g^b)^a$ and Bob computes $k_b = (g^a)^b$. The shared key is thus $k = k_a = k_b \in G$.*

The Diffie-Hellman protocol assumes that even if an adversary knew the generator g and witnessed g^a and g^b, it would be too computationally expensive to compute g^{ab}. Certainly, this seems to be true for large and complex-enough values of d. However, it is currently open whether there exists a efficient algorithm to solve this problem or the related Discrete Logarithm Problem.

Discrete Logarithm Problem: *Let G be a cyclic group and g be a generator of g. Given $h \in G$, find an integer x such that $g^x = h$.*

Clearly, if there exists an efficient algorithm for the Discrete Logarithm Problem, then the Diffie-Hellman protocol is insecure. In the reverse direction, this motivates some ideas of how to select secure groups for the Diffie-Hellman protocol. For instance, the group of integers $\mathbb{Z}$ with generator 1 has a trivial solution for the Discrete Logarithm Problem (where x is the element itself) and so would be a terrible choice for Diffie-Hellman. However, the group of points of a carefully-chosen epileptic curve over a finite field poses an extremely difficult Discrete Logarithm Problem and is widely used for Diffie-Hellman schemes in the real world.

2 Braid Groups in Cryptography

2.1 Defining the Braid Group

The rest of this paper discusses cryptographic schemes based on the braid group. The braid group has unique properties that give way to mathematically hard problems. In this section, we define the group and prove some of its properties. Artin was the first to give a presentation for the group in 1947 [1].

Definition 1. *The **Artin braid group** $\mathbf{B_n}$ for any integer $n \geq 2$ is given by the following presentation:*

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

There is a natural mapping from the braid group B_n to the symmetric group S_n. Recall that S_n is generated by the set of *simple transpositions* $s_1, s_2, \ldots, s_n \in S_n$, where s_i corresponds to the transposition $(i, i+1)$. This set of transpositions also turns out to satisfy the two braid relations. The first relation follows from the general property that any two disjoint cycles in the symmetric group commute. The second relation can be easily verified; for any two neighboring transpositions $s_i, s_j \in S_n$ where $j-i = 1$, we have $s_i s_j s_i = s_j s_i s_j = (i, j, j+1)$.

Lemma 2. *The map $f : B_n \to S_n$ where $f(\sigma_i) = s_i$ is a homomorphism.*

Proof. in Kassel [12]. □

This homomorphism allows us to extend the following property of the symmetric group S_n to the braid group:

Theorem 3. *The group B_n with $n \geq 3$ is non-abelian.*

Proof. The group S_n is non-abelian for $n \geq 3$ because $s_1 s_2 \neq s_2 s_1$. Since the homomorphism $f : B_n \to S_n$ maps to generators of S_n then f is surjective and there exist $\sigma_1, \sigma_2 \in B_n$ that correspond to s_1 and s_2, respectively. By homomorphism, $f(\sigma_1\sigma_2) \neq f(\sigma_2\sigma_1)$ so B_n is non-abelian. □

So far we have defined braids algebraically in terms of generators and relations. However, we can also think of elements of B_n as a *geometric braid* on n strings. The generator σ_i corresponds to the $(i+1)$'th string being crossed over the i'th string (i.e., right over left). It logically follows that the inverse braid σ_i^{-1} corresponds to the i'th string being crossed over the $(i+1)$'th string (i.e., left over right). Below are *braid diagrams* of the geometric braids corresponding to generator σ_i and its inverse.

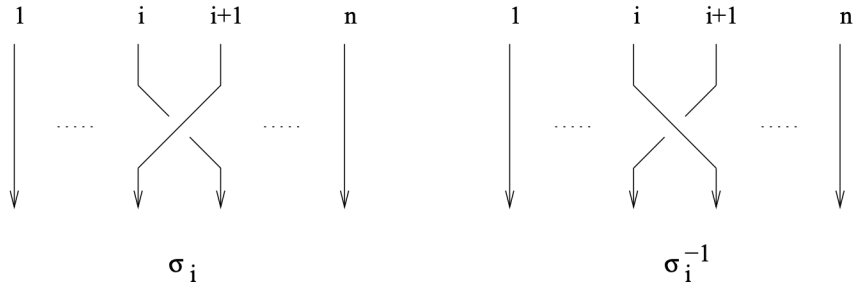

Figure 2: A braid and its inverse [6]

We can verify that our geometric intuition of braid groups matches up to the algebraic group properties.

Proposition 4. *The set of all geometric braids on n strings forms a group.*

Proof. First, note that the operation of crossing strings is well-defined and closed. We now check the axioms.

- It is easy to verify that the operation of crossing strings is associative.
- The identity braid consists of straight strings that are untangled from each other.
- The inverse of a braid is obtained by "flipping" the braid over an imaginary horizontal center axis. Any braid $\beta \in B_n$ can be expressed as a composition of generators $\beta = \sigma_{a_1}\sigma_{a_2}\ldots\sigma_{a_k}$. The inverse of that braid reverses that sequence and takes the inverse of each generator so that the strings cross in the opposite direction. So $\beta^{-1} = \sigma_{a_k}^{-1}\sigma_{a_{k-1}}^{-1}\ldots\sigma_{a_1}^{-1}$.

□

Now examining Artin's relations, the first braid relation states that two geometric braids commute with each other if they do not share a string.

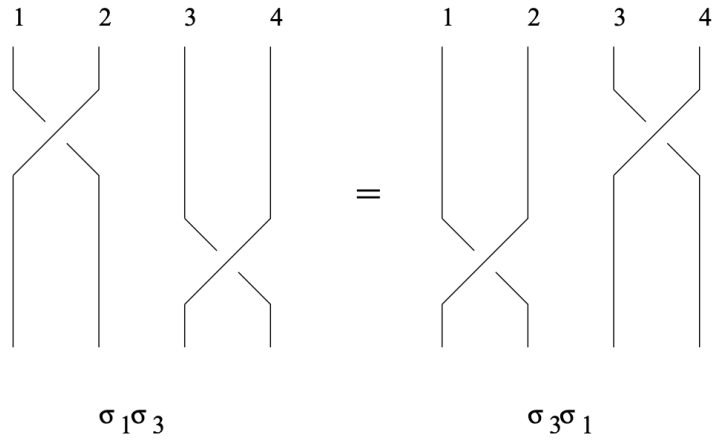

Figure 3: The commutative relation [6]

The second relation is a bit trickier to understand geometrically. It corresponds to the following equality:

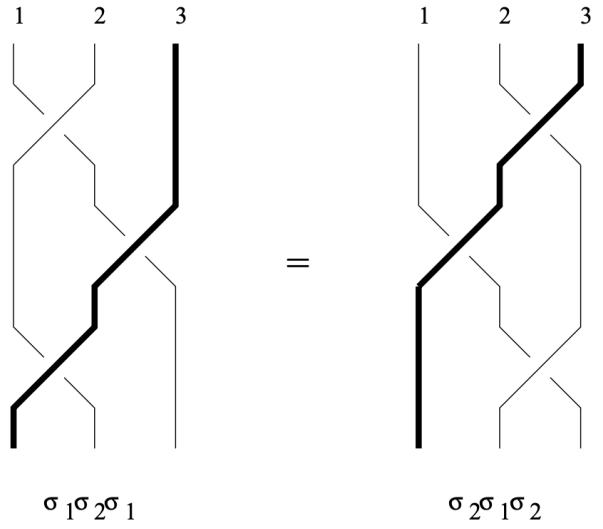

Figure 4: The triple relation [6]

Viewing braids geometrically allows us to conclude the following property of braids that is otherwise nonobvious:

Proposition 5. *Any nonidentity element of B_n has infinite order.*

Consider the composition of generator σ_i with itself. The strings get twisted over one another and never straighten out.

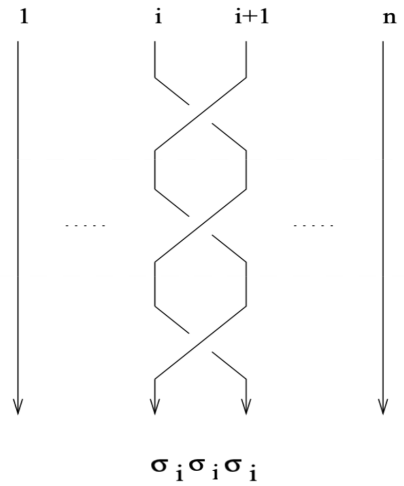

Figure 5: A braid composed with itself

We can also use our geometric intuition to verify the non-abelianness of the braid group. Note that in the below diagram, string 1 crosses two strings on the left but only crosses one on the right.

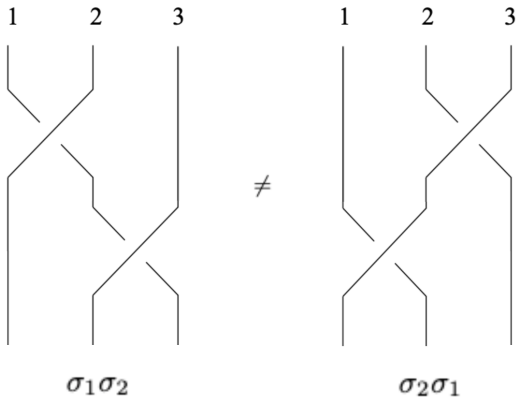

Figure 6: A non-abelian braid operation

Of course, to understand these geometric braid relations, we must define some notion for when two braids are the same. We consider two geometric braids to be identical if they are *homotopic*, meaning that one braid can be continuously deformed into the other. We won't rigorously define homotopy or even geometric braids here, but thinking about elements of B_n geometric helps build some intuition about their behavior. That being said, a natural problem that arises is determining whether two sequences of generators represent the same geometric braid. This is called the *word problem* for B_n. We now present one solution to the word problem, starting with definitions for special braids:

Definition 2. *A braid $\beta \in B_n$ is said to be **positive** if it can be written as a product of generators with positive powers.*

Example 2.1. *The braid $(\sigma_1)^2(\sigma_2)^4\sigma_3$ is positive but the braids $(\sigma_1)^{-1}$ and $\sigma_1(\sigma_2)^{-3}$ are not.*

Definition 3. *The **fundamental braid** of order r is denoted Δ_r and defined as follows:*

$$\Delta_r = (\sigma_1\sigma_2 \dots \sigma_{r-1})(\sigma_1\sigma_2 \dots \sigma_{r-2}) \dots (\sigma_1\sigma_2)(\sigma_1)$$

When considering the group B_n, we often abbreviate Δ_n to Δ.

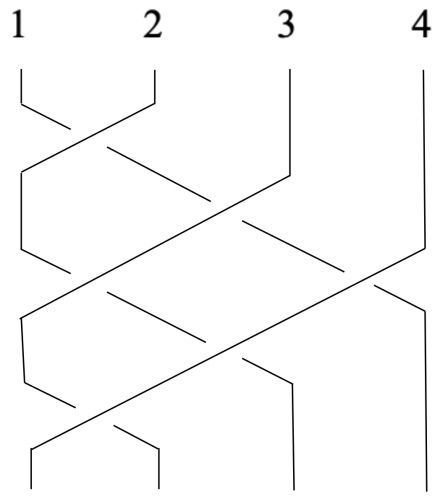

Figure 7: The fundamental braid Δ for the group B_4

Garside's solution to the word problem involves decomposing every sequence of generators into a composition of the fundamental braid raised to some power and a positive braid. He managed to prove that every element in the braid group can be decomposed in such a way.

Theorem 6. *Every element of B_n can be expressed uniquely in the form $\Delta^r P$ where r is an integer, P is a positive braid, and r is maximal for all such representations.*

Proof. in Garside [8]. □

Definition 4. *Any braid in B_n expressed in the unique form $\Delta^r P$ of Theorem 6 is said to be in **normal form**. The index r is called the **power** of the braid.*

The more powerful result of Garside's is that two geometric braids with the same normal form are identical. This result means that we can determine whether two sequences of generators represent the same braid by writing both sequences in their unique normal forms and comparing the two.

Theorem 7. *Two braids in B_n are equal if and only if their normal forms are identical.*

Proof. in Garside [8]. □

2.2 Ko et al. Protocol

Now that we've defined the braid groups and proved some properties, we are ready to examine them in the context of cryptography. Ko et al. proposed a variation of Diffie-Hellman that uses conjugacy properties of the braid group. The protocol is based on the following definition and theorem:

Definition 5. *Let G be a non-abelian group. For $g, h \in G$, we say that h is **conjugate** to g if there exists an element $x \in G$ such that $h = xgx^{-1}$. We express the conjugation of g by x as ${}^{x}g = xgx^{-1}$.*

Theorem 8. *Let G be a non-abelian group and let A, B be subgroups of G such that all elements of A commute with all elements in B. Then, for all $a \in A$, $b \in B$, and $g \in G$, we have ${}^{a}({}^{b}g) = {}^{b}({}^{a}g)$.*

Proof. Writing out the expression for ${}^{a}({}^{b}g)$, we get ${}^{a}({}^{b}g) = a({}^{b}g)a^{-1} = abgb^{-1}a^{-1}$. Since elements of A and B commute, we can express this element as $(ab)g(b^{-1}a^{-1}) = (ba)g(a^{-1}b^{-1}) = {}^{b}(aga^{-1}) = {}^{b}({}^{a}g)$. Thus, ${}^{a}({}^{b}g) = {}^{b}({}^{a}g)$. □

Note that while ${}^{a}({}^{b}g)$ and ${}^{b}({}^{a}g)$ are equivalent expressions for the same element, they may be represented by different sequences of generators. Thus, it may be necessary to rewrite both elements in normal form. This comes up in the last step of the Ko et al. protocol.

Ko–Lee–Cheon–Han–Kang–Park Key Exchange: *Let G be the braid group B_n and g be a publicly-known element of G. Moreover, let ℓ and r be integers such that $\ell + r = n$. We define the following subgroups of G:*

$$A = \langle \sigma_1, \sigma_2, \ldots, \sigma_{\ell-1} \rangle$$

$$B = \langle \sigma_{\ell+1}, \ldots, \sigma_{\ell+r-1} \rangle$$

Now, Alice and Bob execute the following protocol:

1. *Alice selects uniformly at random an element $a \in A$ and stores it. She then computes ${}^{a}g = aga^{-1}$ and sends ${}^{a}g$ to Bob.*
2. *Bob selects uniformly at random an element $b \in B$ and stores it. He then computes ${}^{b}g = bgb^{-1}$ and sends ${}^{b}g$ to Alice.*
3. *Alice computes $k_a = {}^{a}({}^{b}g)$ and Bob computes $k_b = {}^{b}({}^{a}g)$. They use these elements to determine a secret key.*

The braid B_n group is non-abelian and by the first braid relation, all elements of A and B commute with each other. Then, by Theorem 8, the final computed values $k_a = {}^{a}({}^{b}g)$ and $k_b = {}^{b}({}^{a}g)$ are equivalent. So this protocol results in Alice and Bob having a common shared key whereas any adversary only observes the conjugates ${}^{a}g$ and ${}^{b}g$.

This protocol is extremely similar to the Diffie-Hellman protocol laid out at the beginning of this paper. Indeed, it is simply an analog of that protocol where conjugation substitutes for exponentiation. In our discussion of Diffie-Hellman, we defined the Discrete Logarithm Problem as a related problem whose solution would render the Diffie-Hellman protocol insecure. It turns out there is an similar problem for the Ko et al. protocol.

2.3 Conjugacy Search Problem

Ko et al. is just one example of a whole family of cryptographic schemes that depend on the conjugacy of braid groups. These protocols are hinged on the hardness of the Conjugacy Search Problem. Formally, we define the problem as follows:

Conjugacy Search Problem: *For a non-abelian group G, let $g, h \in G$ where $h = {}^x g$ for some $x \in G$. Given g and h, find an element $y \in G$ such that ${}^y g = h$.*

Garside gave the first solution to the conjugacy problem for the braid group B_n. I outline his proof here, but the full details are left to Garside's original paper [8]. We start off with some definitions.

Definition 6. *The **word-length** of an element $\beta \in B_n$ is denoted $L(\beta)$ and is defined as the number of generators in its representation.*

Example 6.1. *The braid $(\sigma_1)^2(\sigma_2)^4\sigma_3$ has word-length 7. The braid $(\sigma_1)^{-2}\sigma_2$ has word-length 3.*

Definition 7. *The **index-length** of an element $\beta \in B_n$ is the least number of strings needed to make a closed braid representation.*

Example 7.1. *The braid $(\sigma_1)^{-1}\sigma_2(\sigma_3)^2$ has index-length 2. The below figure helps show this. The standard braid diagram is on the left and the closed braid representation with two strings is on the right. The closed braid representation is created by connecting the endpoints of the strings at each index back to the starting string at that index. In the diagram, the two strings in the closed braid are distinctly colored red and blue and the original braid is in bold. Note that the added connections do not cross over each other or any other strings.*

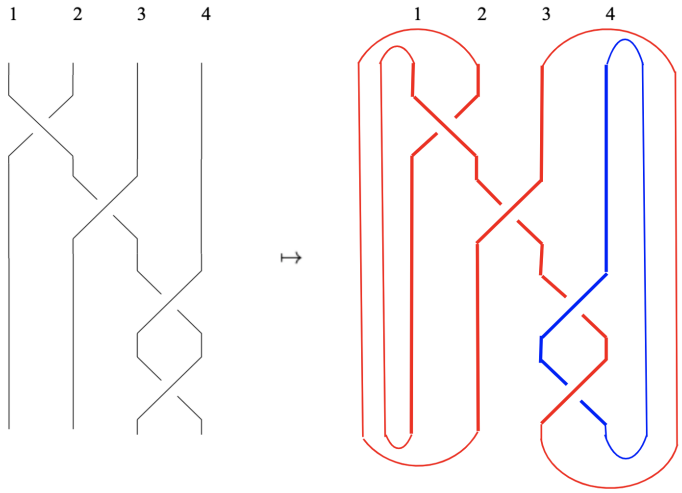

Figure 8: Diagram of $(\sigma_1)^{-1}\sigma_2(\sigma_3)^2$ and its closed braid representation

We can use these definitions to define an important lemma [8]. Recall that the *power* of a braid is the value of r when the braid is expressed in normal form $\Delta^r P$.

Lemma 9. *In B_n, the number of braids in normal form of index length t and power $\geq r$ is finite.*

Proof. Let $\Delta^m P$ be any word satisfying the conditions. Then, if $L(\Delta) = d$, we have

$$m \geq r \tag{1}$$

$$t = md + L(P) \tag{2}$$

Since $L(P) \geq 0$ and d is positive, the last equation gives us

$$m \leq \frac{t}{d} \tag{3}$$

Equations 1 and 3 bound m and show that the number of values of m is finite. Equation 2 shows that for any fixed m, $L(P)$ is constant so the number of possible values of P is finite. This proves the given claim. □

Garside's solution involved constructing subsets of B_n. These sets are constructed from an initial element $\beta \in B_n$. We conjugate β by a set of braids called *α-transformations* that are specific to that value of β. The products are elements of the form $\alpha\beta\alpha^{-1}$ which we add to our set. These elements all have the property that they have the same index length as β and that they are all conjugate to β. We repeat this process for each distinct element in our set, conjugating it by the α-transformations. By Lemma 9, this set of elements is finite so we eventually reach a point where further applications of this process do not yield new braids. The set being finite additionally implies that there is some maximal power reached by elements of the set. We use this power to define a smaller set of elements:

Definition 8. *The **summit set** of a braid β is the subset of elements of maximal power contained within the set constructed by the described process.*

Garside's construction of summit sets allowed him to prove a powerful result that solves the conjugacy problem.

Theorem 10. *For two elements $\beta, x \in B_n$, x is conjugate to β if and only if their summit sets are identical.*

Proof. in Garside [8]. □

The main issue is that the summit sets of a given braid may be exponentially large and thus take exponential time to construct. An attack on a braid-based cryptographic scheme using this approach is then still infeasible. There have been several attempts to bound the size of the summit set or define alternate constructions, but none yet have been successful in yielding a polynomial time algorithm. However, this does not mean that the braid-based protocols are secure. In the following section, we will highlight methods of attacking these cryptosystems that arise from properties of the braid group.

2.4 Attacks Using Heuristic Algorithms

With cryptography, unlike in mathematics, one does not need an efficient solution that works in all cases in order to break a braid-based cryptosystem. Often, it suffices to attack a cryptosystem with an *heuristic algorithm* that works in a large proportion of cases. Hofheinz and Steinwandt were among the first to suggest an algorithm that solves the conjugacy search problem in braid groups [9]. Their algorithm is based on the observation that representative of conjugate braids in summit sets tend to be conjugate by a *permutation braid* which is easy to guess. The algorithm was able to successfully break a braid-based Diffie-Hellman-type scheme in 70% to 80% of cases. We wrap up this paper by briefly mentioning two other types of heuristic algorithms that are the subjects of much ongoing research. Some papers are referenced if the reader is interested in learning more about these algorithms.

2.4.1 Linear Algebra Attacks

This type of attack involves taking a matrix representation of the braid group and solving the conjugacy search problem using linear algebra. There are two well-known representations of the braid group. The first is the Bureau group, which maps the braid generators to the matrix:

$$\sigma_i \mapsto \left(\begin{array}{c|ccc} I_{i-1} & 0 & 0 & 0 \\ \hline 0 & 1-t & t & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & I_{n-i-1} \end{array}\right)$$

where I_k denotes the $k \times k$ identity matrix. Hughes [11] and Lee and Lee [15] provide two such algorithms using the Bureau representation.

The problem with the Bureau representation is that it is unfaithful for $n \geq 5$. The Lawrence-Krammer representation on the other hand is faithful for all n. Its matrices are too complicated to write out here, but one algorithm that uses the Lawrence-Krammer representation is Cheon and Jun's polynomial-time attack [3].

2.4.2 Probabilistic Length-Based Attacks

This type of attack requires a length function $l : B_n \to \mathbb{Z}$ that maps braids to some integer value. For example, the function could output the word-length of a braid's normal form representation. Now, for an instance of the Conjugacy Search Problem, we are given $\beta, x \in B_n$ and want to find $y \in B_n$ so that ${}^y\beta = x$. If we can write $y = y'\sigma_i$ where y' is shorter than y, then for all $j \neq i$, it should be the case that $l(\sigma_i y^{-1} x y \sigma_i^{-1}) < l(\sigma_j y^{-1} x y \sigma_j^{-1})$. This fact allows us to "guess" the value of i by checking the lengths of the braids. After finding σ_i, we have a shorter y' that we can repeat the process on. Eventually, we will guess enough generators to determine the value of y.

The effectiveness of this attack relies on the chosen length function. There are several length functions that have been shown to work with high probability for the braid group. For more details and examples of this type of attack, refer to Hughes and Tannenbaum [10] and Garber et al. [7].

3 Conclusion

The challenge of cryptography is defining schemes that are easy for the communicating parties to execute but difficult for an adversary to decipher. The braid group then is an excellent basis for cryptographic schemes. It is non-abelian and has no efficient solution to the Conjugacy Search Problem, yet at the same time it can be entirely defined by two relations and has an easy-to-compute solution to the word problem. Moreover, we saw in this paper how looking at the braid group both algebraically (in terms of generators) and geometrically (in terms of strings) inspires new ways of understanding the group elements and their interactions with each other. Some properties become obvious when viewing braids as twisted strands while others are more readily defined in terms of composed generators. In all, the complex nature of the braid group gives rise to many interesting applications and computationally hard problems.

These topics remain an active area of research in cryptography. Specifically, there is much work being put into finding a polynomial-time solution to the Conjugacy Search Problem. There have been several refinements to Garside's summit set method over the years (see the *super summit set* [5] and the *reduced super summit set* [14]) but a fast algorithm remains elusive. It is possible in the near future that we may see advances in the Conjugacy Search Problem that bring us closer to a polynomial-time solution. While such a solution would deal a blow to cryptography by rendering braid-based protocols insecure, this knowledge would also motivate more-secure cryptographic schemes based on harder problems in group theory.

4 Bibliography

[1] Emil Artin, The theory of braids, *Annals of Math.* **48** (1947) 101-126.

[2] Blackburn et al., Group Theory in Cryptography (2010), `https://arxiv.org/pdf/0906.5545.pdf`.

[3] Jung Hee Cheon and Byungheup Jun, A polynomial time algorithm for the braid Diffie-Hellman conjugacy problem, in *Advances in Cryptology – CRYPTO 2003* (D. Boneh, ed), Lecture Notes in Computer Science **2729** (Springer, Berlin, 2003) 212–225.

[4] Whitfield Diffie and Martin E. Hellman, New directions in cryptography, *IEEE Trans. Information Theory* **22** (1976) 644–654

[5] E. A. Elrifai and H. R. Morton, Algorithms for positive braids, *Quart. J. Math. Oxford* **45**, No. 2 (1994), 479-497.

[6] David Garber, Braid group cryptography, in *Braids: Introductory Lectures on Braids, Configurations and Their Applications* (J. Berrick, F.R. Cohen, E. Hanbury, eds) (World Scientific, Singapore, 2009).

[7] David Garber, Shmuel Kaplan, Mina Teicher, Boaz Tsaban and Uzi Vishne, Probabilistic solutions of equations in the braid group, *Adv. Appl. Math.* **35** (2005), 323–334.

[8] F. A. Garside, The braid group and other groups, *Quart. J. Math.* Oxford **20** (1969), 235–254.

[9] D. Hofheinz and R. Steinwandt, A practical attack on some braid group based cryptographic primitives, in *Public Key Cryptography – PKC 2003* (Y.G. Desmedt, ed.), Lecture Notes in Computer Science **2384** (Springer, Berlin, 2002), 176–189.

[10] James Hughes and Allen Tannenbaum, Length-Based Attacks for Certain Group Based Encryption Rewriting Systems (2002), `https://arxiv.org/PS_cache/cs/pdf/0306/0306032v1.pdf`.

[11] James Hughes, A linear algebraic attack on the AAFG1 braid group cryptosystem, in *Information Security and Privacy* (G. Goos, J. Hartmanis and J. van Leeuwen, eds), Lecture Notes in Computer Science **2384** (Springer–Verlag, Berlin, 2002), 176–189.

[12] Christian Kassel and Vladimir Turaev. *Braid groups*, volume 247 of *Graduate Texts in Mathematics*. Springer, New York, 2008.

[13] Ki Hyoung Ko, Sang Jin Lee, Jung Hee Cheon, Jae Woo Han, Ju-sung Kang, and Choonsik Park, New public-key cryptosystem using braid group, in *Advances in Cryptology - CRYPTO 2000* (M. Bellare, ed.), Lecture Notes in Computer Science **1880** (Springer, Berlin, 2000) 166–183.

[14] Sang Jin Lee, Algorithmic solutions to decision problems in the braid groups, Ph.D. thesis, Korea Advanced Institute of Science and Technology, 2000.

[15] Sang Jin Lee and Eonkyung Lee, Potential weaknesses of the commutator key agreement protocol based on braid groups, in *Advances in Cryptology – EUROCRYPT 2002*, (L. Knudsen, ed.), Lecture Notes in Comp. Science **2332** (Springer, Berlin, 2002) 14–28.