SOME COMBINATORIAL ASPECTS OF BRAID GROUPS AND THEIR APPLICATIONS IN CRYPTOGRAPHY

Ivanovo - Russia, March 21, 2012

Daniel Tieudjo
Max Planck Institute for Mathematics
Bonn - Germany

tieudjo@mpim-bonn.mpg.de
tieudjo@yahoo.com

Motivations:

Braids are familiar since ancient times

- Decoration, Communication, Personalization, Events, ….

Braid groups have been studied from different point of view

- Topologically and Geometrically since 1925 when they were introduced by E. Artin.

- Algebraically since first results appeared

- Cryptographic applications

Braid groups attract more attention for their use in pubic-key cryptography, first by Anshel, Anshel and Goldfeld in 1999:

I. Anshel, M. Anshel et D.Goldfeld : An algebraic method for public key cryptography, Math. Research Letters 6 (1999) 287–291.

Since then, there has been intensive research in this field: new public-key schemes were developed and broken.

Aim:

- Review some algorithmic and combinatorial properties of braid groups,
- Show how braid groups are applied in cryptography
- Some discussions

OUTLINE:

I. Overview on braid and braid groups

II. Some combinatorial properties of braid groups

III. Overview on Cryptography

IV. (Basic) Braid based Public Key Cryptosystems (PKC) and some (major) attacks

V. Discussions on the future of the Braid Group Cryptography (BGC)

I. Overview on braid and braid groups

I.1- Braids

One can imagine braid (group) from an geometric approach.

A braid = A sequence of crossed (twisted) strands.

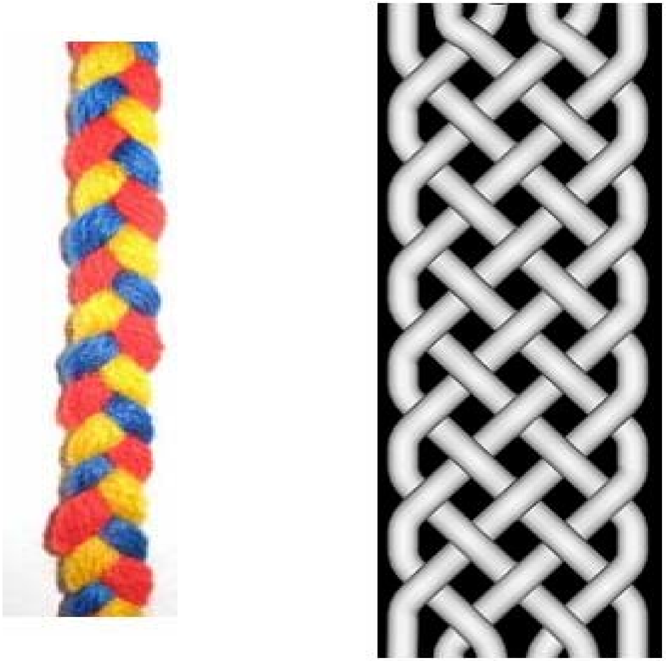


Fig. 1.1: Decorative braids

Indeed, braids are three-dimensional figure consisting of n disjoint curves connecting the points (1,0,0), …, (n,0,0) to the points (1,0,1), …, (n,0,1) in R^3.

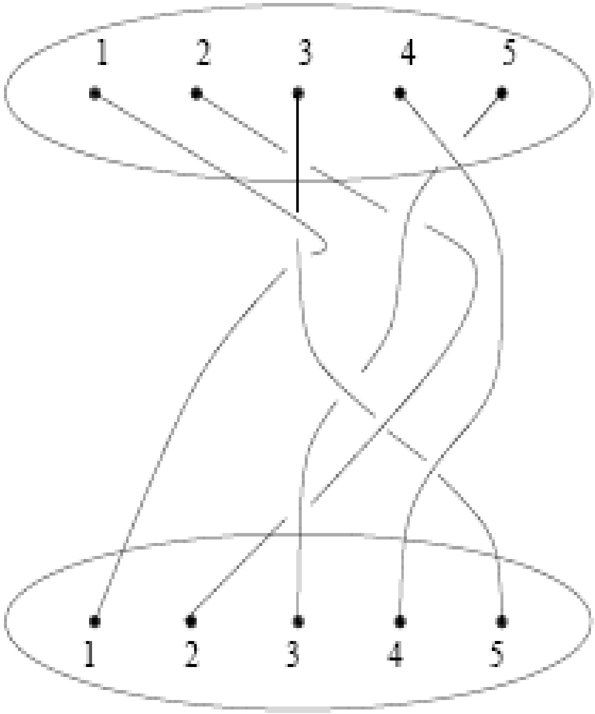
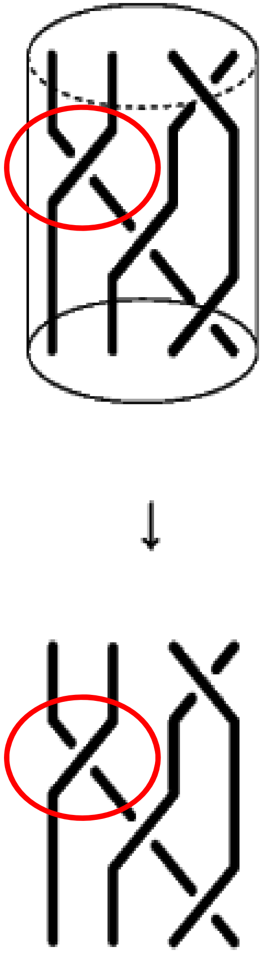


Fig. 1.2: A braid on 5 strands

Planar diagrams are just the planar projection of the 3-dimensional figure. So, it is important to specify, when two strands intersect, which crosses over the other.



Thus a braid can be associated to a (**at least one**) planar diagram.

Braids are usually considered to start at the top and end at the bottom.

Fig. 1.3: From braid to braid diagram

Intuitively, strands are labeled by 1, 2, to n, attached on 2 parallel horizontal bars.

An n-braid is obtained by intertwining the strands and fixing the lower ends on the lower bar. Notice that a pair of strands can be intertwined in two ways: by passing the strand on the left over or under the strand on the right.

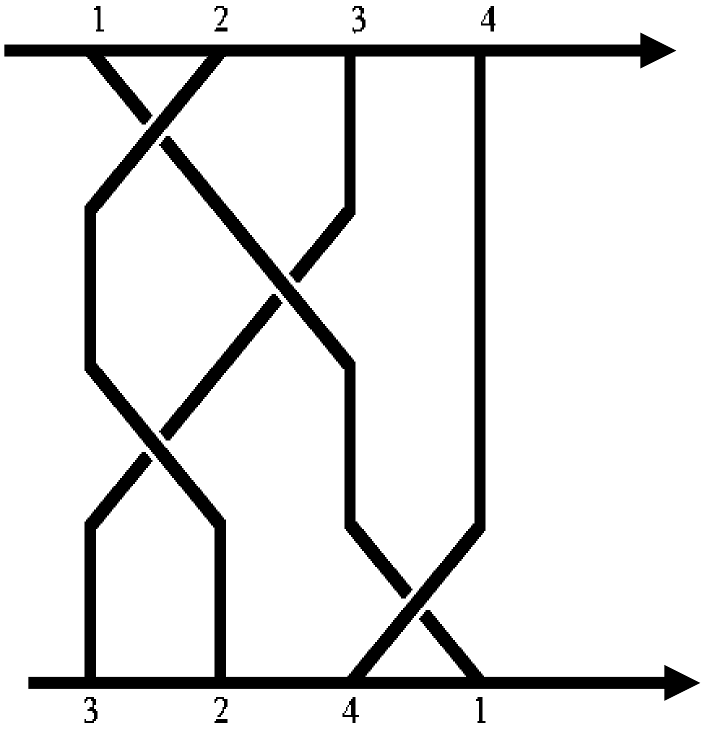


Fig. 1.4: A braid diagram

One could continue the twisting arbitrarily many times, or twist in the reverse direction.

I.2- Particular braids

Trivial braid denoted 1 = all strands run parallel with no crossings.

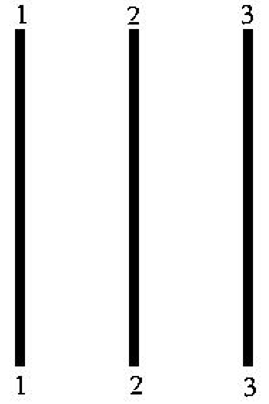


Fig. 1.5: the trivial braid on 3 strands diagram

Elementary braids σ_i : strands i and i+1 intersect; strand i+1 crosses over strand i, and other strands run parallel with no crossings.

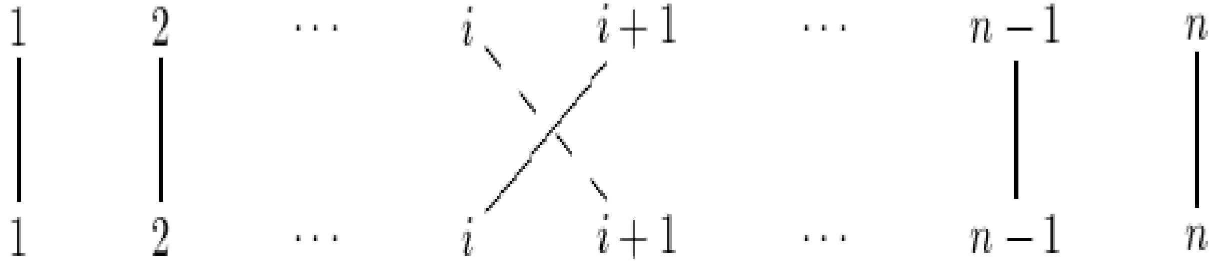


Fig. 1.6: σ_i diagram

Braid a_{sr} : represents the braid formed by lifting strands s and r $(s > r)$ above all the others, crossing strand s over strand r, and then setting the strands down again.

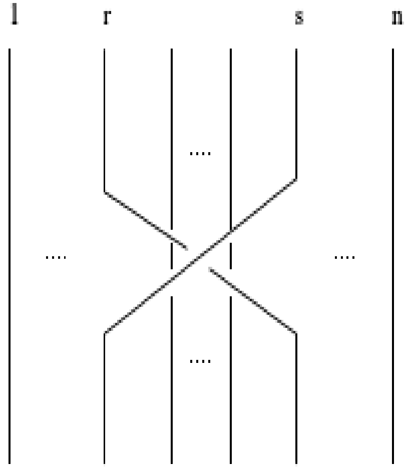


Fig. 1.7: a_{sr} diagram

Observe the similitude with permutations !!! But the main difference : how strands cross ?

Fundamental braid Δ_n: represents the braid on n strands, where any two strands cross positively exactly once.

When all the strands cross positively, we called the braid *positive braid*. *Permutation braids* are positive braids such that each pair of strands crosses at most once.

So Δ_n is a permutation braid defined by: $i^\Delta = n - i + 1$ $(1 \leq i \leq n)$

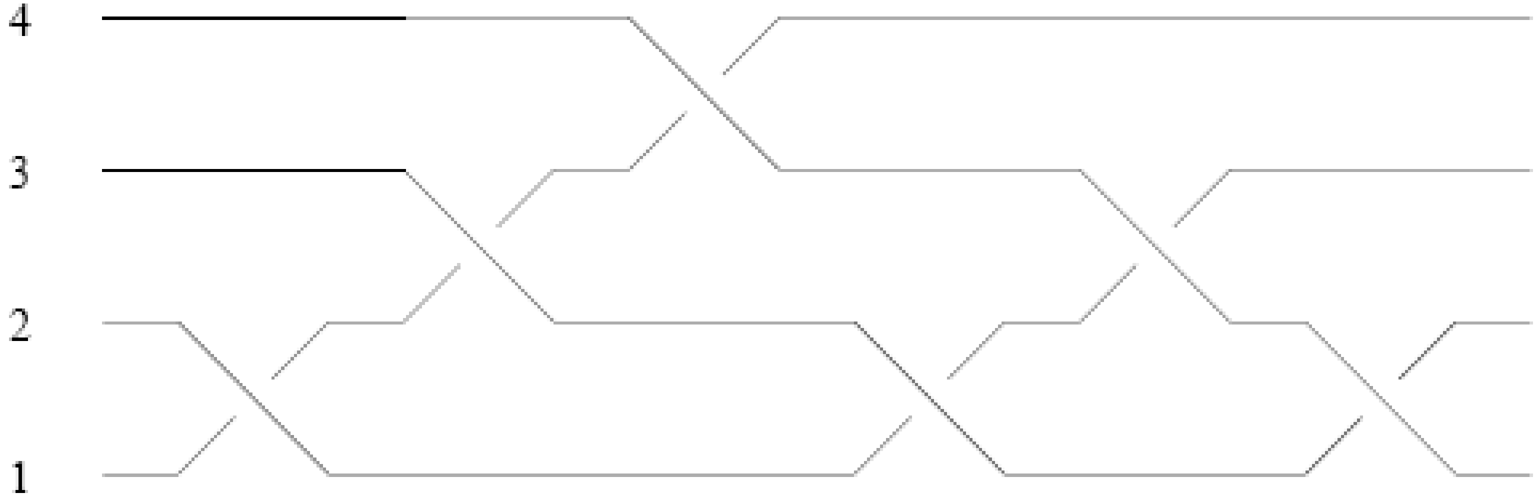


Fig. 1.8: The fundamental braid Δ_4 on 4 strands diagram

I.3- Composition (product) of two braids (braid diagrams) =

concatenation (juxtaposition)

i.e. match up the ends of the strands on the first braid to the beginnings of strands on the second braid.

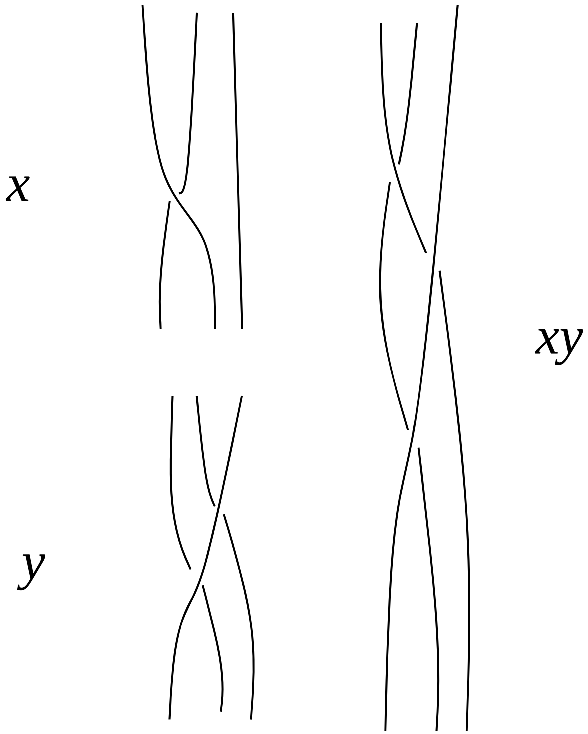


Fig. 1.9: The product of two braid on 3 strands

Now

- This product is associative;

- The product of an n-strand braid by the trivial n-strand braid will give the n-strand braid. So, **the trivial n-strand braid behaves like the identity element**.

Let D_n be the set of n-strand braid diagrams.

Then D_n with the above defined product is a **monoid**.

Attention! In a planar diagram, some intersections of strands can be independent on each other; i.e. a braid can be represented by several planar diagrams.

I.4- braid diagrams equality = continuous deformation (diagram isotopy)

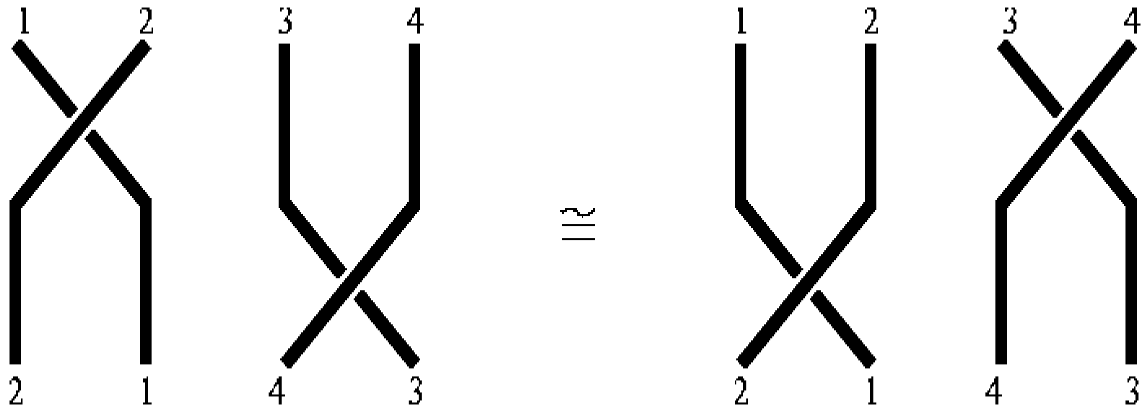
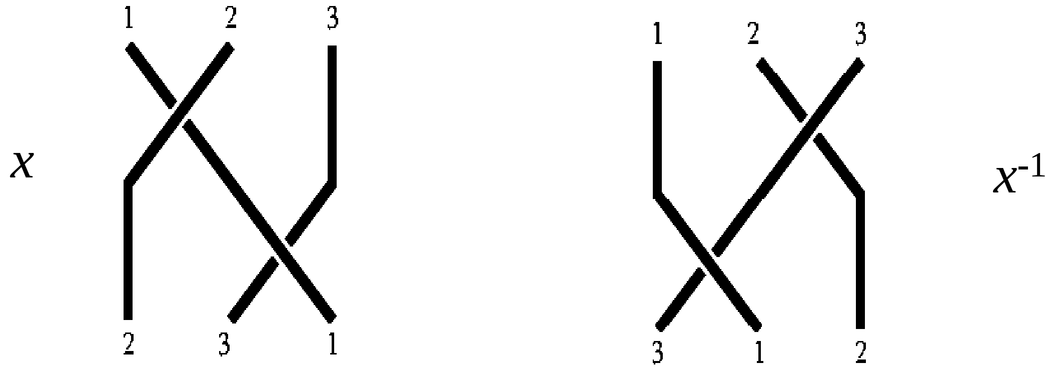


Fig. 1.10: Equal (equivalent) braids

Two braids are considered *equal* if one can be obtained from the other by sliding the crossings past one another without adding or removing any other crossings and without cutting the strands.

The relation « ≅ » diagram isotopy is an **equivalent relation** which is **compatible** with product (concatenation).

I.5- Braid inverse = The inverse of any braid is its mirror image with the face of the mirror perpendicular to the strings.



Indeed

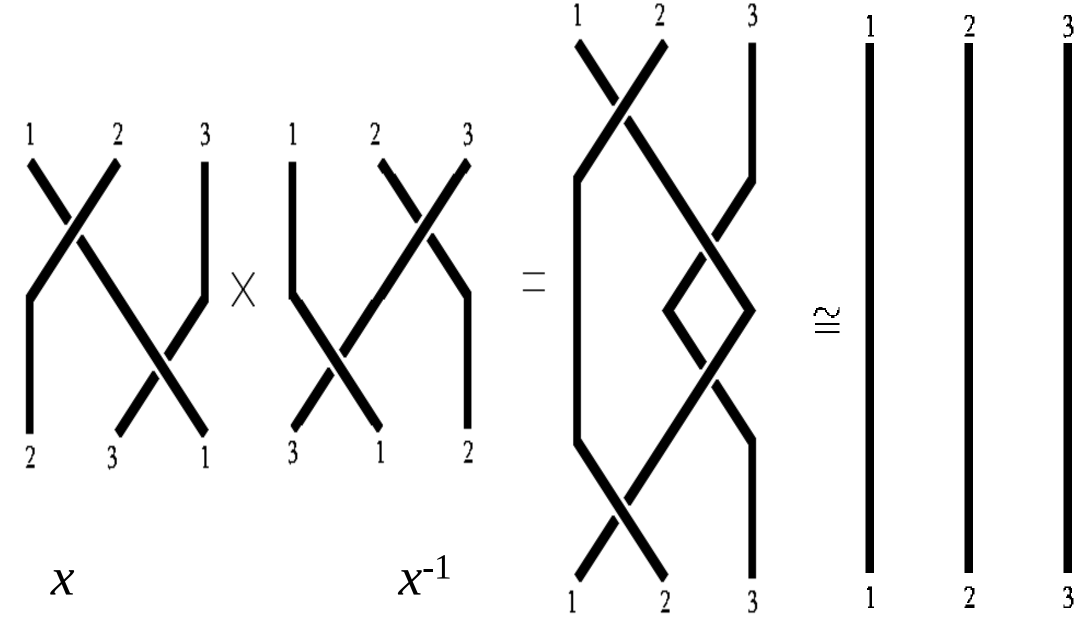


Fig. 1.11: The inverse of a braid

I.6- Braid group

Let $\boldsymbol{B_n} = \boldsymbol{D_n}/\cong$

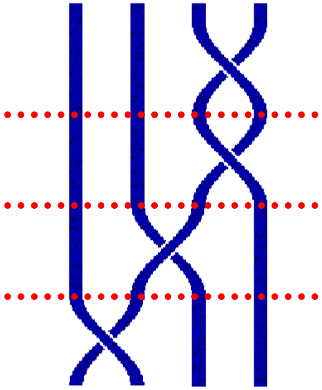
Then $\boldsymbol{B_n}$ with the product (concatenation) is a group.

This is called the n-strands **braid group**.

braid group.= ({braid diagrams, up to isotopy}, concatenation)

Remarks

Let x be the following braid. It is possible to cut into small parts corresponding to elementary braids



$$\sigma_3^{-2}\sigma_2\sigma_1^{-1}$$

Fig. 1.12: Decomposition of braid through elementary braids

Any braid can be written as a product of elementary braids together with their inverses

i.e $B_n = grp(\sigma_1, \ldots, \sigma_{n\text{-}1})$

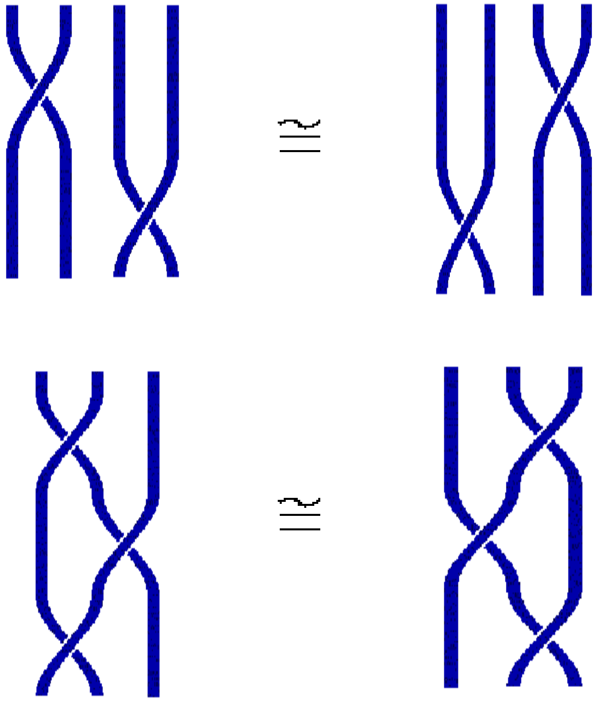
The elementary braids ($\sigma_1, \ldots, \sigma_{n\text{-}1}$) are the call the ***Artin generators***.

- In fact, E. Artin (1925) proved that the only relations involving these generators are:

$$\sigma_i \sigma_j = \sigma_j \sigma_i \qquad |i - j| > 1$$

$$\sigma_i \sigma_j \sigma_i = \sigma_j \sigma_i \sigma_j \qquad |i - j| = 1$$

Illustrated by:



$$\sigma_i \sigma_j = \sigma_j \sigma_i$$

$$\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$$

Fig. 1.13: Braid group relators diagrams

So, the n-strands braid group is given by generators and relators:

$$B_n = \left\langle \sigma_1, \ldots, \sigma_{n-1} \ : \ \begin{array}{ll} \sigma_i\sigma_j = \sigma_j\sigma_i & |i-j| > 1 \\ \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j & |i-j| = 1 \end{array} \right\rangle$$

There exist several presentation of the Braid Group B_n

For example through the ***Birman-Ko-Lee generators*** a_{sr} [Garber, 2009 or Dehornoy, 2006]

Many different approaches to define Braid groups provide its various properties and tools to prove and solve Problems : combinatorial, geometric, topological approaches, …

II. Some combinatorial properties of braid groups

II.1. From an algebraic point of view

- B_1 = {1} is the trivial group. $B_2 = \langle\sigma_1\rangle$ is isomorphic to Z;

- $B_3 = \langle\sigma_1,\sigma_2 : \sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2\rangle \cong \langle x,y : x^2 = y^3\rangle$ where $x = \sigma_1\sigma_2\sigma_1$, $y = \sigma_1\sigma_2$.

- $B_4 = \langle\sigma_1,\sigma_2,\sigma_3 : \sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2 , \sigma_2\sigma_3\sigma_2 = \sigma_3\sigma_2\sigma_3 , \sigma_1\sigma_3 = \sigma_3\sigma_1 \rangle$

- Z(B_n), the centre of B_n , is the infinite cyclic subgroup Z(B_n) = $\langle\Delta_n^2\rangle$, where

$$\Delta_n = (\sigma_1 \cdot\cdot\cdot \sigma_{n\text{-}1})(\sigma_1 \cdot\cdot\cdot \sigma_{n\text{-}2}) \cdot\cdot\cdot \sigma_1 ;$$

Indeed, in B_n one has: $\Delta_n\sigma_i = \sigma_{n\text{-}i}\Delta_n$, and thus $\Delta_n^2\sigma_i = \sigma_i\Delta_n^2$, for $1 \le i \le n - 1$.

- $(B_n)^{ab}$ = Z, the group of integers.

- B_n contents the symmetric group S_n of n elements;

- B_n is not abelian for $n > 2$; (Take $\sigma_1\sigma_2 \neq \sigma_2\sigma_1$ as permutations).

- If we add the relations $\sigma_i^2 = 1$ for all possible i, we get the presentation of S_n . Hence, S_n is a quotient of B_n.

- The map $\pi : B_n \to S_n$ such that $\sigma_i \mapsto \tau_i = (i, i+1)$ is an epimorphism. Its kernel is the *pure braid group* denoted $\ker\pi = P_n$.

- B_n is embeddable in B_{n+1} (natural inclusion) ; Thus $B_1 \subseteq \cdots \subseteq B_n \subseteq B_{n+1} \subseteq \cdots$

Let $B_\infty = \bigcup\limits_{n\geq 1} B_n$. B_∞ is the direct limit of the B_n .

II.2. From a geometric point of view

B_n is the fundamental group of certain configuration space (Fenn, 1999; Rolfsen, 2008)

For example, B_3 is the fundamental group of the complement of the trefoil knot $K \subset S^3$. i.e. $B_3 \cong \pi_1(S^3 - K) \cong \pi_1[SL(2,\mathrm{R})/SL(2,\mathrm{Z})]$.

- B_n is torsion free (Gonzales-Meneses, 2011),
- Braid groups are linear see (Bigelow, 2001) and (Krammer, 2000).
- ……

II.3. Residual properties

$B_n \leq \mathrm{Aut}(F_n)$, where F_n is the free group on n symbols $x_1, \ldots, x_n$.

Indeed, for any $1 \leq i \leq n - 1$, define σ_i by :

$$x_i \mapsto x_{i+1} \,,$$

$$x_{i+1} \mapsto x_{i+1}^{-1} x_i x_{i+1} \,,$$

$$x_j \mapsto x_j \;\; for \;\; j \neq i, i+1.$$

Obviously any σ_i ($1 \le i \le n - 1$) is an automorphism of Aut(F_n).
So $gp(\langle\sigma_1, \dots, \sigma_{n-1}\rangle) \le$ Aut(F_n). Now, $B_n \to$ Aut(F_n) is injective and $B_n \le$ Aut(F_n).

- B_n , P_n are residually finite, thus Hopfian. $\mathrm{Aut}(B_n)$ is residually finite.

- P_n is residually torsion-free nilpotent (see Bardakov & Bellingeri, 2007)

- B_n is not subgroup separable for $n \ge 4$. see (Dasbach & Mangum, 2000)

However B_n is π_c (cyclic subgroup separable). see (E. Feder, 2009) .

Consequence : WP is solved, but the GWP - not

- $\mathrm{Aut}(B_n) = \langle \mathrm{Inn}(B_n), \varepsilon \rangle$ where $\varepsilon : \sigma_i \mapsto \sigma_i^{-1}$ and is ***complete*** for $n \ge 4$ (centreless and characteristic). see (J. Dyer and E. Grossman, 1981).

Problems : What about p-residallity of B_n , P_n , $\mathrm{Aut}(B_n)$, for some prime p ?

Are B_n , P_n , $\mathrm{Aut}(B_n)$ residually nilpotent ?

What about conjugacy subgroup (or p) separability in B_n , P_n ?

II.4. Algorithmic properties and problems

- The WP is *effectively* solved *in polynomial time* in Braid groups (there exist canonical, normal forms for each element of the group)
  - *Garside normal form :* $w = \Delta_n^r P_1 P_2 \cdots P_k$:
  - *Birman-Ko-Lee canonical form :* $w = \delta_n^j A_1 A_2 \cdots A_k$:
  - *Dehornoy handle reduction .*

There exist polynomial algorithms and computer platforms to represent a braid on a normal form, and this representation is unique.

- *Difficult problems* in Braid groups
  - CP is very difficult to solve when $n > 6$. (CDP, CSP, MCSP)
  - RP is very difficult (RP, REP). (Tsaban, 2007)
  - Decomposition problem,
  - Triple Decomposition problem,
  - Twisted Conjugacy Problem (TCP) (Gonzales-Meneses, 2011)
  - Schifted Conjugacy Problem (ShCP, ShCSP) (Longrid & *al*. 2009)
  - …
- Residuality, hyperbolicity and small cancellation problems ?
- New generators and consequences ?

Why Braid groups are so interesting ?

Despite their theoretical role in various domains (mathematics (topology, geometry, …) physics, biology, …)

- Computability (arithmetic of braid groups)
- Existence of ‘’ difficult ‘’ problems

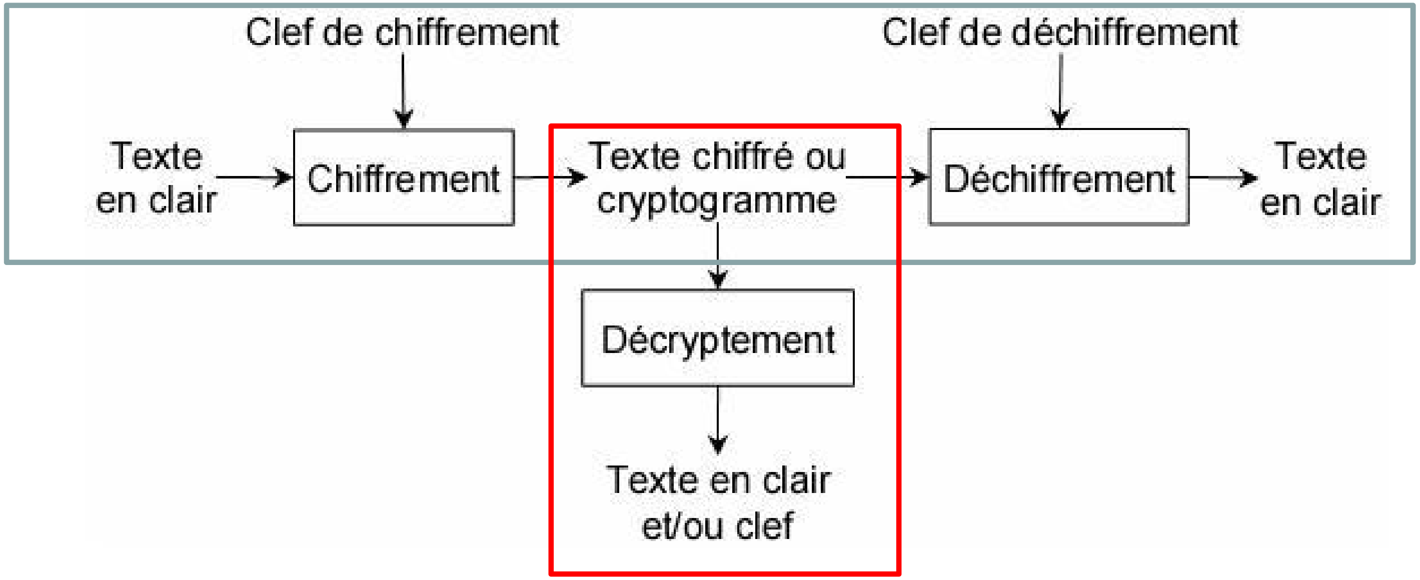
	Numbers	Braids
Divisibility (*gcd*, *lcm*)	✓	✓
Order	≤	✓ (in some sense)
Primality	prime	Simple (permutation braid)
Decomposition (WP)	Factorization	Normal forms
Power	DLP	CP (CSP)
	DH	DH
	RP	RP (REP)

There exist packages to compute braids (MAGMA, CBRAID, CRAG, SINGULAR, GAP, BRAIDING, SAGE, …).

In this last decade, Public Key Cryptosystems based on the CP in braid groups have been developed and broken. The intensive research on this field leads to the so-called « **Braid Group Cryptography (BGC)**».

III. Overview on Cryptography

III.1- Cryptology = Cryptography + Cryptanalysis



Cryptography = secured data transmission, exchange.

For message transmission, the plain text is protected by a transformation (**encryption**). The scripted text (non-comprehensible) obtained is retransformed (**decryption**) to obtain the plain text.

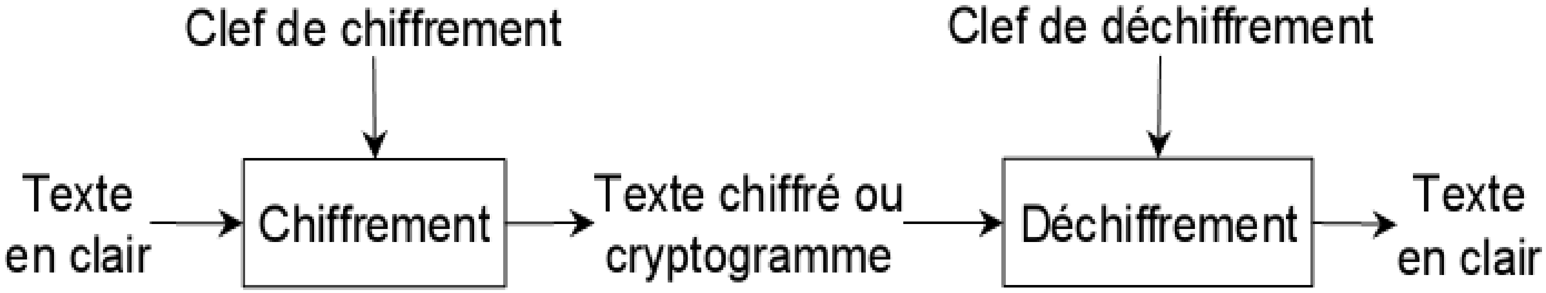
Cryptanalysis = vulnerabilities of algorithms needed to secure data transmission, exchange.

Cryptosystems or cryptographic algorithms are mathematical functions needed to encrypt and decrypt.

III. 2. Symmetric Cryptography and Public Key Cryptography

Symmetric Cryptography :

When the encryption and the decryption keys can be deduced one from the other, we talk about Symmetric Cryptography.



The well known protocols are DES, 3DES, AES, …

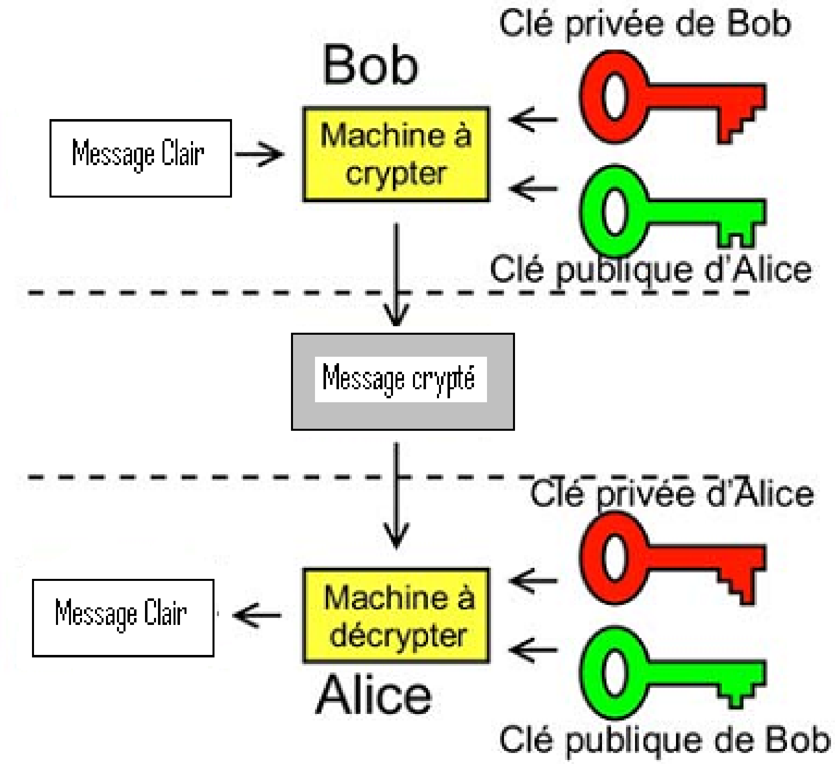
Advantage: There are fast.

Problem : But how do you get the Keys ?

Public Key Cryptography (Diffie-Helman, 1976)

In this case, the users has 2 keys:
- 1 key known as public key, largely diffused
- 1 key known as private key, which is kept secret

These systems are called **asymmetric or Public Key Cryptosystems (PKC)**



PKC lie on some functions called « one way functions ».

PKC has been applied for:

- Confidential message transmission;

- Key exchange (KEP or KAP);

- Authentication ;

- Signature.

Alice sends to Bob a (clear or ciphered) message with a signature proving the origin of the message.

Note that each signature scheme leads to an authentification scheme.

- ….

III-3 Examples of some PKC

RSA

FP: Find two large primes p and q, each about 100 digits long. Let $n = pq$ and $k = \varphi(n) = (p-1)(q-1)$, the Euler number. Choose a random integer r [$3 \leq r \leq k$] such that r has no common factors with k. It is easy to find D (the inverse of r modulo k), that is $Dr \equiv 1 \pmod{k}$.

Public key	r and n
Private key	p, q, k, D
Bob wants to send a plain message text m to Alice ($m \in Z_n$)	
Enciphering (Bob)	$c = m^{r} \pmod{n}$
Deciphering (Alice)	$m = c^{D} \pmod{n}$

This method is currently secure, since in order to determine the secret decryption key D, the intruder should factor the number n (200 or so digit), which is a very hard task.

El Gamal

DLP: find a such that $g^a \equiv A$ $(mod\ p)$; p is a big prime, g and A are given, g of order p–1 modulo p and p does not divide A with $1 \le a \le p-2$.

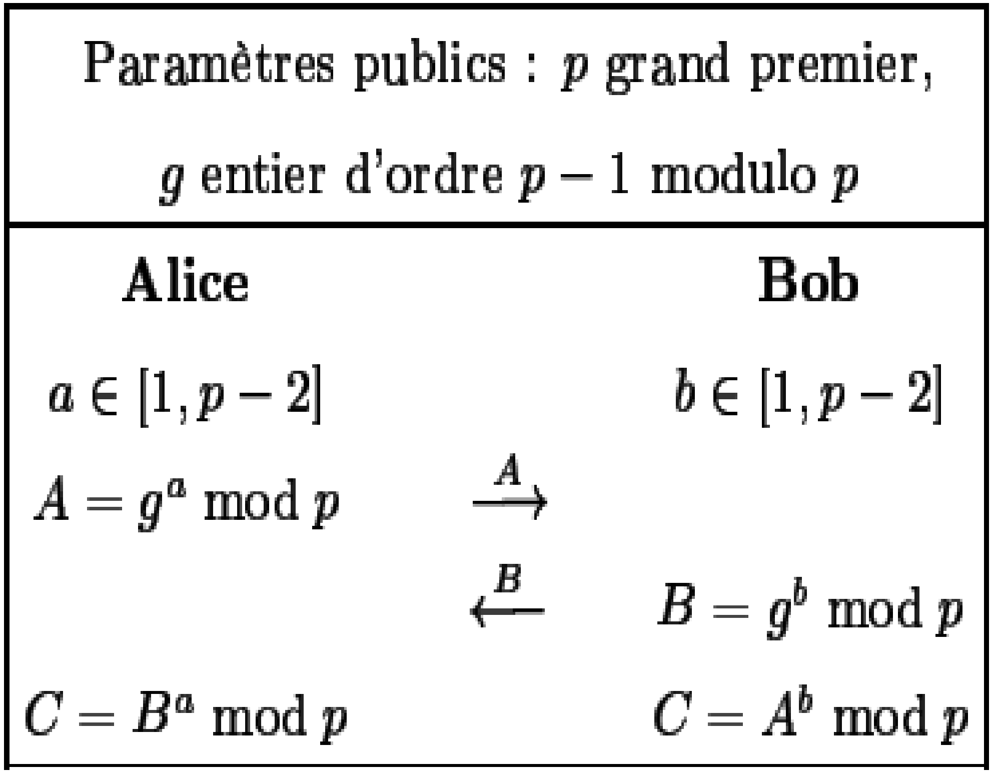
Public key	p, g and A
Private key	a
Bob wants to send a plain message text m to Alice ($\mathrm{m} \in \mathrm{Z}_p$)	
Enciphering (Bob)	Chooses k, $1 \le k \le p-2$, and computes $K = g^k$ (mod p) Computes $\mathrm{c} = \mathrm{m}A^k$ (mod p) and sends (K,c) to Alice
Deciphering (Alice)	$\mathrm{m} = \mathrm{c}K^{-a}$ (mod p)

Indeed, $\mathrm{c}K^{-a} \equiv \mathrm{c}g^{-ak} \equiv \mathrm{c}A^{-k} \equiv \mathrm{m}$ (mod p)

DH KEP

DHP: find C, such that $C = g^{ab}$ $(mod\ p)$, a and b are unknown integer such that $g^a \equiv A$ and $g^b \equiv B$ $(mod\ p)$; p is a big prime, g, A and B are not divisible by p, with g of order p–1 modulo p.

DH KEP: Two persons (Alice and Bob) want to share a **common secret key**, which they will use for message transmission (through a classical cryptographic system).



Paramètres publics : p grand premier,

g entier d'ordre $p-1$ modulo p

Alice		**Bob**
$a \in [1, p-2]$		$b \in [1, p-2]$
$A = g^a \bmod p$	$\xrightarrow{A}$	
	$\xleftarrow{B}$	$B = g^b \bmod p$
$C = B^a \bmod p$		$C = A^b \bmod p$

Fiat-Shamir Authentication Scheme

Authentication: Alice (the prover) wishes to prove her identity to Bob (the verifier) i.e. she wishes to prove that she knows some private (secret) key without enabling an intruder watching the communication to deduce anything about her private key.

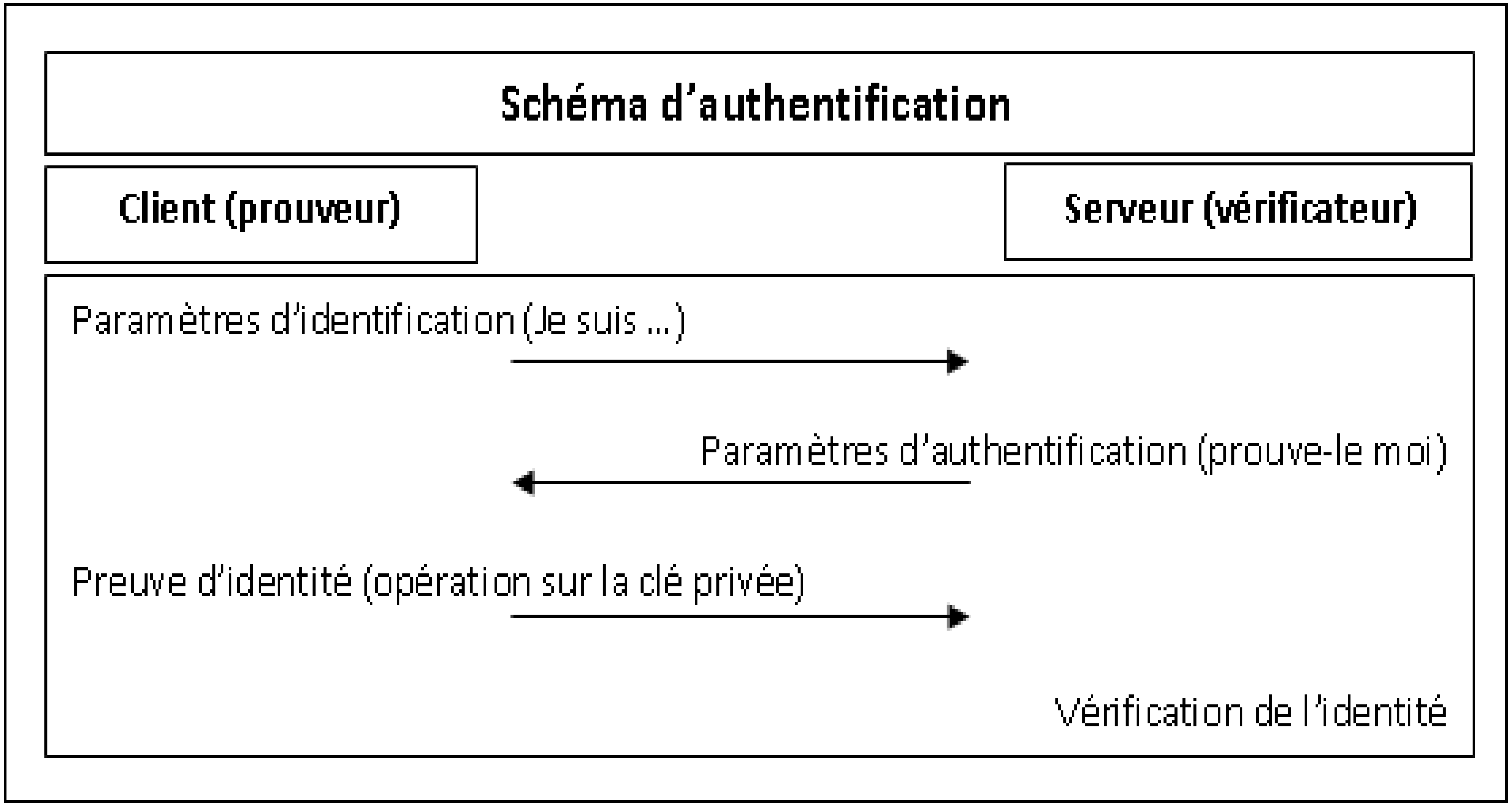


Schéma d'authentification

Client (prouveur)

Serveur (vérificateur)

Paramètres d'identification (Je suis ...)

Paramètres d'authentification (prouve-le moi)

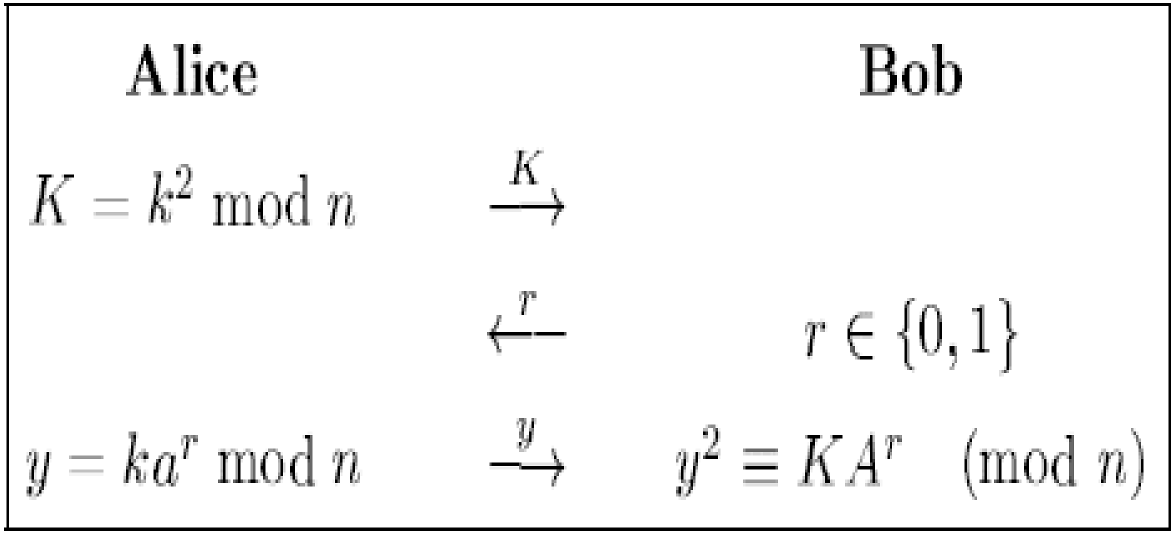
Preuve d'identité (opération sur la clé privée)

Vérification de l'identité

Fiat-Shamir Authentication Scheme based on RSA and Rabin

FP and SQRP: Find two large primes p and q, each about 100 digits long. Let $n = pq$ Alice chooses an integer A in $[1, n\text{-}1]$ such that $A \equiv a^2 \bmod n$.

Public Key	n , A
Private key (Alice)	a
Alice whishes to be identified by a server or by Bob	
Engagement (Alice)	Chooses k, computes $K = k^2 \bmod n$ and sents K
Challenge (Server or Bob)	Takes $r \in$ {0,1} and sends
Answer (Alice)	Computes $y = ka^r$ mod n and sends
Verification (Server or Bob)	Computes y^2 and compare with KA^r mod n



Alice		Bob
$K = k^2 \bmod n$	$\xrightarrow{K}$	
	$\xleftarrow{r}$	$r \in \{0,1\}$
$y = ka^r \bmod n$	$\xrightarrow{y}$	$y^2 \equiv KA^r \pmod{n}$

Remark : RSA, El Gamal , DH KEP , Fiat-Shamir protocols above are based on difficult problems (Factorization, DLP, DHP, …) in **number theory**

IV. Braid based Public Key Cryptosystems and Majors attacks

IV.1 Basic Cryptosystems based on braid groups

We now present some basic PKC based on braid groups. We will present:

- The AAG KEP;

- The KLCHKP KEP or DH type KEP,

- A transmission scheme (El Gamal-type)

- A Fiat-Shamir type Authentication scheme

Nearly all these schemes are based on the difficulty to solve the **CP** in braid groups.

CSP: Given two braids p, p' which are conjugate. Find the conjugator i.e. an element s which satisfies: $p' = s^{-1}ps$.

So, Assuming that the CSP is difficult enough in Braid groups, we consider two public sets of braids. Alice and Bob who want to share a common secret key and they have a private key each.

The AAG KEP (AAG, 1999)

Public keys	2 sets of braids: $p_1,\ldots, p_k$ and $q_1,\ldots q_m$ in B_n		
	ALICE		**BOB**
Private keys	a word u on k letters and their inverses		a word v on m letters and their inverses
Action	$s = u(p_1,\ldots, p_k)$ $q'_i = sq_is^{-1}$ $(i=1,\ldots, m)$ →		$r = v(q_1,\ldots, q_m)$ ← $p'_j = rp_jr^{-1}$ $(j=1,\ldots, k)$
Secret key	$K_A = su(p'_1,\ldots, p'_k)^{-1} = [s, r]$		$K_B = v(q'_1,\ldots, q'_m)r^{-1} = [s, r]$

where $u(p_1,\ldots, p_k)$ is the substitution of the i-th letter of the alphabet by p_i (for all $1 \leq i \leq k$).

Indeed, $K_A = su(p'_1,\ldots, p'_k)^{-1} = sru(p_1,\ldots, p_k)^{-1}r^{-1} = srs^{-1}r^{-1}= sv(q_1,\ldots, q_m)s^{-1}r^{-1}$ $= v(q'_1,\ldots, q'_m)r^{-1} = K_B$.

The AAG KEP is based on the **MCSP**, a variant of the CP.

The KLCHKP KEP or DH-type (KLCHKP, 2000)

Although braid groups are not commutative, they contains large subgroups such that each element of the first subgroup commutes with each element of the second subgroup

Note $\boldsymbol{B_n}$ the n-strand braid group and $\boldsymbol{B_{n,2n}}$ the subgroup of B_{2n} generated by $\sigma_{n+1}, \ldots, \sigma_{2n-1}$.

Public keys	one braid p in B_n		
	ALICE		**BOB**
Private keys	a braid a in B_n		a braid b in $B_{n,2n}$
Action	$p_A = apa^{-1}$ →		← $p_B = bpb^{-1}$
Secret key	$K_A = ap_Ba^{-1}$		$K_B = bp_Ab^{-1}$

Remark: The braids a and b commute since they are words of strands 1 to n and n+1 to 2n respectively. So $ab = ba$ and then $ap_Ba^{-1} = abpb^{-1}a^{-1}$= $bapa^{-1}b^{-1} = bp_Ab^{-1}$.

The security here is based on the difficulty to find a from (p, p_A) and/or b from (p, p_B); the CSP (DH-type KEP).

El Gamal on braid groups (KLCHKP, 2000)

Bob wishes to send Alice a message m. He can use Alice's public key to encipher his message. Alice must be able to retrieve Bob's message using her private key, but an intruder watching the communication should not.

Public key	p, p' with $p' = sps^{-1}$, $p, p' \in B_n$
Private key	s, $s \in LB_n$
Bob wants to send a plain message text m to Alice (m $\in \{0,1\}^N$)	
Enciphering (Bob)	Chooses a random $r \in UB_n$ and sends the encrypted text m'' = m $\oplus$ $h(rp'r^{-1})$ together with additional datum $p'' = rpr^{-1}$.
Deciphering (Alice)	m = m'' $\oplus$ $h(sp''s^{-1})$.

where $LB_n = \langle \sigma_1, \cdots, \sigma_{n-1} \rangle$ and $UB_n = \langle \sigma_{n+1}, \cdots, \sigma_{2n-1} \rangle$ and h is collision free one-way hash function.

Indeed, the braids s and r commute; so $sp''s^{-1} = srpr^{-1}s^{-1} = rsps^{-1}r^{-1} = rp'r^{-1}$; thus m'' $\oplus$ $h(sp''s^{-1})$ = m $\oplus$ $h(rp'r^{-1}) \oplus h(sp''s^{-1})$ = m.

Fiat-Shamir type Authentication on braid groups (SDG, 2006)

The proover (Alice) chooses a braid $t = s^m \in B_n$, for some integer m, publishes t and keep s secret.

Public Key	t , m
Private key (Alice)	s
Alice whishes to be identified by a server or by Bob	
Engagement (Alice)	Chooses $r^m \in B_n$, computes $x = rtr^{-1}$ and sents x .
Challenge (Server or Bob)	Takes $k \in \{0,1\}$ and sends
Answer (Alice)	If k = 0, sends $y = r$ If k = 1, sends $y = rsr^{-1}$
Verification (Server or Bob)	If k = 0, checks that $yty^{-1} = x$ If k = 1, checks that $y^m = x$

Broken [Groch & *al*., 2006]

Some more cryptosystems based on braid groups

There exist some more cryptographic schemes on braid groups

- **Group Authentification schemes:**

  Group Authentication schemes based on CSP and the Root Extraction Problem were proposed by H. Sibert, P. Dehornoy & M. Girault, (2006).

- **Signature schemes:**

  Signature schemes based on Matching Conjugate Search Problem (MCSP) were proposed by K.H. Ko & al. (2002).

- **Group Signature schemes:**

  Group Signature schemes based on CSP, DP and RP were proposed by T. Thomas and A:K/ Lal in 2006.

- **…**

IV.2 Some major attacks

As we see, these basic cryptographic schemes in braid groups are based on the CP (MCSP, CSP, CDP …). The security of these schemes depends on the difficulty to solve the CP in braid groups. So the proposed attacks are algorithms that attempt to solve this problem.

Several strategies are considered. Here we will sketch majors ones, namely:

- *-SS attacks (algorithms) and the heuristic algorithm (solution of the CP);
- Linear representation attack (using auxiliary groups);
- Length based attacks (probabilistic approach).

Remark that there exist some more attacks.

IV.2.1 *SS attacks [S.J. Lee and E. Lee, 2002]

The approach: given $x \in B_n$.

x is associated with a distinguished finite set $SS(x)$ of its conjugates.

($SS(x)$ called the Summit Set).

Then this set can be replaced with some of its subsets ($SSS(x)$ or $USS(x)$ or $\mathrm{RSS}(x)$) which is smaller and therefore easier to determine.

Given $x \in B_n$, compute a finite subset $SS(x)$ of the conjugacy class of x such that:

(1). $x, y \in B_n$ are conjugate iff $SS(x) = SS(y)$
(2). For each $x \in B_n$, one can compute efficiently a representative $\bar{x} \in SS(x)$ and an element $a \in B_n$ such that $a^{-1}xa= \bar{x}$
(3). There is a finite algorithm which can construct the whole set $SS(x)$ for any representative $\bar{x} \in SS(x)$.

Now, solve the CDP/CSP for a given x , y by performing the following steps:

(a). Find representatives $\overline{x} \in SS(x)$ and $\overline{y} \in SS(y)$
(b). Using the algorithm from (3), compute further elements of $SS(x)$ (while keeping track of the conjugating element), until either:

(ii) $\overline{y}$ is found in $SS(x)$, so x and y are conjugate and the conjugating element is provided, or
(ii) the entire set $SS(x)$ has been constructed without encountering $\overline{y}$, and x and y are not conjugated.

Different algorithms are based on this approach and the computation of special subsets $SS(x)$, $x \in B_n$.

$\boldsymbol{SSS(x)}$ = finite set of the conjugates of x having minimal canonical length $len(x)$ (or having maximal infimum and minimal supremum, at the same time)

Given $x \in B_n$, one can find $\overline{x} \in SSS(x)$ by a finite sequence of special conjugations called **cycling** and **decycling** (*see D. Gaber or S.J. Lee and E. Lee*)

$$x = \Delta^p x_1 \cdots x_r \in B_n \qquad r > 0$$

$$c(x) = \Delta^p x_2 \cdots x_r \tau^{-p}(x_1) = \left(\tau^{-p}(x_1)\right)^{-1} x \left(\tau^{-p}(x_1)\right)$$

$$d(x) = x_r \Delta^p x_1 x_2 \cdots x_{r-1} = \Delta^p \tau^{-1}(x_r) x_1 x_2 \cdots x_{r-1} = x_r^{-1} x x_r$$

$$\tau : \sigma_i \mapsto \sigma_{n-i} \qquad 1 \le i \le n$$

Now, if $x, y \in B_n$ and the representatives $\overline{x} \in SSS(x)$ and $\overline{y} \in SSS(y)$ are computed, one can check whether x and y are conjugate:

- If $inf(x) \neq inf(y)$ or $sup(x) \neq sup(y)$, then x and y are not conjugate.

- Otherwise, compute $SSS(x)$ and check whether $\overline{y} \in SSS(x)$].

Remark: the cycling function c maps $SSS(x)$ to itself. So define

$SSS(x) \supseteq \boldsymbol{USS(x)}$ = consisting of all $y \in SSS(x)$ such that $c^m(y) = y$, for some $m > 0$. Thus, $\boldsymbol{USS(x)}$ consists of a finite set of disjoint orbits, closed under cycling.

Now using the $\boldsymbol{USS}$, one can solve the CDP/CSP in braid groups.

(*see D. Gaber or S.J. Lee and E. Lee for details*).

A heuristic algorithm using SSS [Hofheinz and *al*., 2002]

The idea:

If we start with two elements in the same conjugacy class, their representatives in the *SSS* will be conjugated by a permutation braid.

So, given a pair (x , x') of braids, where $x' = s^{-1}xs$, we do the following steps:

(1). By a variant of cycling and decycling, we find $\overline{x} \in SSS(x)$ and $\overline{x}' \in SSS(x)$;

(2). Try to find a permutation braid P such that $P^{-1}xP= \overline{x}$‘ .

If P is found, then at the end we will have at hand the needed conjugator for breaking the cryptosystem, since the conjugators can be followed in the cycling/decycling process.

Reduction of the CSP: [Maffre, 2005 & 2006]

IV.2.2 Linear attacks

The method here is to use linear representations of braid groups: i.e. mapping braid groups into an auxiliary group (of matrices), in which the CSP is easy. In this way, one can solve the CSP in B_n , since the CP is easy in linear groups.

The best known linear representation of the braid group B_n is the Colored Burau representation [H.R. Morton, 1999]

The Colored Burau Group:

For i = 1, …, n, let $y_i = \left(C_i(t_i), \quad (i\ i+1)\right)$

where (i i+1) is a transposition (when $i = n$ the transposition is defined to be (i 1))

and
$$C_i(t) = \begin{pmatrix} 1 & & & & & \\ & 1 & & & & \\ & & t & -t & 1 & \\ & & & & & \\ & & & & & 1 \end{pmatrix}$$

with 1 on the diagonal, 0 elsewhere, except in the i-th row where we have **0 0** ··· **0** ***t*** **–*t*** **1 0** ··· **0 0** and –t on the diagonal.

Now, denote $\boldsymbol{CB_{n+1}}$ = gp$\langle y_1, \cdots , y_n\rangle$.
An element of $\boldsymbol{CB_{n+1}}$ is of type **(M, σ)** where M is an n x n matrix with coefficients that are finite Laurent polynomials in the variables $t_1, \ldots , t_n$ over the integers, and σ is a permutation in S_{n+1} , and the product in $\boldsymbol{CB_{n+1}}$ is defined by

$$(M, \sigma) \cdot (M', \sigma') = (M \text{ x } \sigma(M'), \sigma\sigma'),$$

where **$\sigma(M')$** denotes the matrix obtained from M' by permuting the variables $t_1, \ldots, t_n$ appearing in the coefficients of M' by σ.

One can check that the elements y_i ($i = 1, \ldots n-1$) satisfy the braid relations.

So, we have a homomorphism from B_n to CB_n. Thus every braid can be associated to a colored Burau matrix.

The attack [J. Hugues, 2002]:

- Take one or several pairs of conjugate braids (p, p') associated with the same conjugating braids.
- Compute their Burau image
- Solve the CSP (MCSP) in the linear group.

Since the kernel of this representation is “small”, there is a non-negligible probability that we find the correct conjugator and hence we break the cryptosystem.

Remarks:

1. S.J. Lee and E. Lee (2002) indicated a weakness in the AAG protocol based on the shared key.

2. The **Lawrence-Krammer representation** (D. Krammer, 2002):
This is another linear representation which associates every braid in B_n with a matrix of size $\binom{n}{2}$ with entries in a 2-variable Laurent Polynomial ring $Z[t^{\pm 1}, q^{\pm 1}]$

$$B_n \to GL_{\binom{n}{2}}\left(Z\left[t^{\pm 1}, q^{\pm 1}\right]\right)$$

Using this representation, J. Cheon and B. Jun (2003) attack the DH-type protocol.

IV.2.3 Some more attacks

Length-based attacks (probabilistic heuristic approach)

Idea (J. Hugues and A. Tannenbaum, 2000):

Retrieve a conjugator for a pair (p , p') by starting with p', which is supposed to be derived from p, then iteratively conjugate p' into a new braid $tp't^{-1}$ of minimal length, and finally check whether $tp't^{-1}$ is equal to p .

For more details, see D. Gaber, S. Kaplan, B. Tsaban and U. Vishne (2005, 2006)

Some more attacks exist (see D. Gaber or P. Dehornoy).

V. Future direction of the BGC

We now discuss future direction for the BGC. Some ideas for the construction of new cryptosystems.

V.1. A cryptosystem based on the Shifted Conjugacy Problem (ShCP) [Dehornoy, 2006]

We know that B_n is embeddable in B_{n+1} ; Thus $B_1 \subseteq \cdots \subseteq B_{n\text{-}1} \subseteq B_n \subseteq B_{n+1} \subseteq \cdots$

Let $B_\infty = \bigcup_{n\geq 1} B_n$

Definition: Let $x, y \in B_\infty$. We define $x * y = x \cdot dy \cdot \sigma_1 \cdot dx^{-1}$

where dx is the shift of x in B_∞ i.e. $d : \sigma_i \mapsto \sigma_{i+1}$ for each $i \geq 1$.

Problem (ShCSP): Let $s, p \in B_\infty$ and $p' = s * p$. Find a shifted conjugator i.e. a braid $\tilde{s}$ satisfying $p' = \tilde{s} * p$

P. Dehornoy proposed a scheme like the Fiat-Shamir authentication scheme. So let S be a set with a binary operation which satisfies:

$$r * (s * p) = (r * s) * (r * p)$$ (S is an LD-system)

The protocol is as follows:

Alice is the prover who wants to convince Bob that she knows the secret key s.

Protocol:

Public key:	Two elements $p, p' \in S$ such that $p' = s*p$.
Private keys:	Alice: $s \in S$
Alice:	Chooses a random $r \in S$ and sends Bob $x = r*p$ and $x' = r*p'$.
Bob:	Chooses a random bit c and sends it to Alice.
Alice:	If $c = 0$, sends $y = r$ (then Bob checks: $x = y*p$ and $x' = y*p'$); If $c = 1$, sends $y = r*s$ (then Bob checks: $x' = y*x$).

Now one can use the shifted conjugacy operation as the $*$ operation on B_∞ in order to get a LD-system.

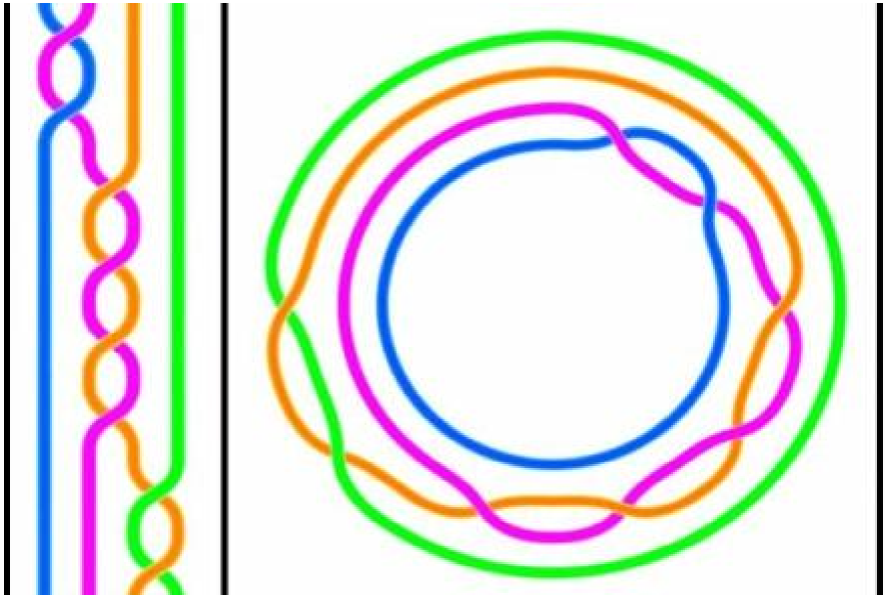
Problems:

- cryptanalysis direction (SS theory, length based attack) ? Yes ! [Longrid & al., 2009]

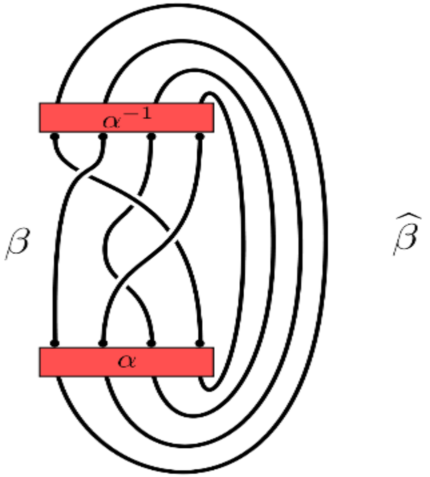
- cryptosystem direction (LD-system on the braid group or different group) ? Yes ! [Kalka, in progress]

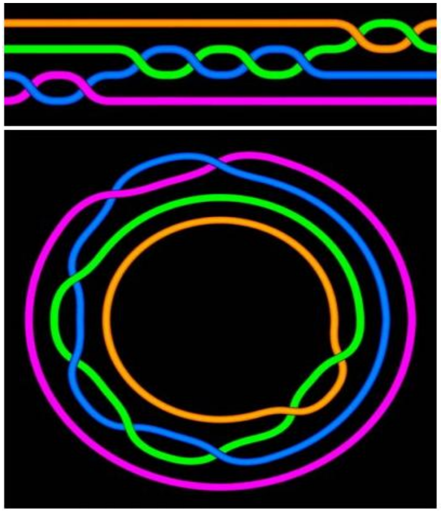
V.2. Link to knot theory: the Markov chains [J. Birman (2002), Glez Vasco (2003)]

One of the reasons for the interest of braid groups is the relation with knot theory. A knot can be tough as the **closure** of a braid i.e. consider the 3-dimensional braid diagram; joint the i ends (top and bottom) from the planar diagram projection to obtain an oriented knot.



2 braids which represent the same element in the group will represent the same knot; and also distinct braids can represent the same knot (for example conjugate braids).





Problem: Can one decide when 2 braids will represent the same knot ? [see, J. Birman for an affirmative answer called the Markov theorem]

Definition: (Elementary) Markov movements M_1 and M_2

$$b \xrightarrow{M_1} aba^{-1} \quad a,b \in B_n \quad ; \; b \xrightarrow{M_2} b\sigma \quad \text{where } \sigma = \sigma_n^{\pm 1}$$

Markov movements = finite applications of the elementary Markov movements $b = b_0 \to b_1 \to \cdots \to b_{m-1} \to b_m = b'$

Theorem (Markov Theorem): 2 braids (not necessary with the same number of strands) will represent the same knot iff one can be obtained from the other by a Markov movement

Markov movements = equivalence relation
The problem of finding the Markov chains generalizes the CP in braid groups.

Now it is possible to build a KEP based on this problem

KEP based o Markov chains:

Let M_A and M_B be two set of Markov sequences such that $[M_A , M_B] = 1$ and one can decide whether for any $M_a \in M_A$ and $M_b \in M_B$, if the Markov sequences M_aM_b and M_bM_a generate exactly the same transformation on a braid.

Public keys	2 sets of Markov sequences: M_A and M_B and a braid b in B_n		
	ALICE		**BOB**
Private keys	a sequence M_a in M_A		a sequence M_b in M_B
Action	$b_1 = M_a(b)$ → ←	→ ←	→ $b_2 = M_b(b)$
Secret key	$K_A = M_\mathrm{a}(b_2)$		$K_B = M_\mathrm{b}(b_1)$

Problem: How to design the sets M_A and M_B such that the result would be more secure than the KLCHKP protocol ?

V.3. Pure braids, combed braids and combing braids

S.S. Magliveras, D.R. Stinson and T. Trung (2002) proposed the scheme MST_1 based on the so-called **logarithmic signature** for finite groups.

Let G be a permutation group of order n. A logarithmic signature is some sequence of the form $\alpha = [\alpha_1, \dots , \alpha_s]$, where α_i is a finite sequence in G (i.e. a sequence of elements of S_n), with $|G| = \Pi len(\alpha_i)$.

Every element of the group has a unique representation as a product of LS; WP or Factorization Problem (FP)

The security of MST_1 is based on the construction of the logarithmic signature. The idea above is generalized for infinite groups and a construction of logarithmic signature on pure braid groups [Glez Vasco, (2003)]

It is possible to write each pure braid as a product of so called « **combed braids** » i.e. pure braids such that all but one strand are trivial (or the first n-1 strands are parallel).

Let $w \in P_n$ in the Artin generators. w can be uniquely written as: $w = v_1 \cdots v_{n-1}$

where all the v_i are combed braids. This leads to an algorithm to solve the WP in the pure braid groups.

MST$_1$ cryptosystem

Let G be a group of permutations of order n, let η be a ***supertame*** logarithmic signature (the factorization can be achieved in time $O(n^2)$.

Public keys: G, η and a pair of logarithmic signature (α,β) with α ***wild*** (not tame) and β ***tame*** (the factorization can be computed in polynomial time of order n.

Private key: $\{\theta_1, \ldots , \theta_k\}$ transversal logarithmic signature such that

$$\hat{\beta}^{-1}\hat{\alpha} = \hat{\theta}_1 \cdots \hat{\theta}_k$$

Encryption: Let $m \in Z_{|G|}$ be the message; the scripted message is

$$c = \hat{\beta}^{-1}\hat{\alpha}(m)$$

Decryption: The plain text is obtained by computing:

$$m = \hat{\alpha}^{-1}\hat{\beta}(c) = \hat{\theta}_k^{-1} \cdots \hat{\theta}_1^{-1}(c)$$

Every logarithmic signature α induces a bijection $\hat{\alpha} : Z_{|G|} \to G$

There exists algorithms for combing braids [R.S.D. Thomas (1971)].

Let $A_{i,j} = \left(\sigma_{j-1}\sigma_{j-2}\cdots\sigma_{i+1}\sigma_i^2\sigma_{i+1}^{-1}\cdots\sigma_{j-2}^{-1}\sigma_{j-1}^{-1}\right)$

We have $P_n = \left\langle A_{i,j} \right\rangle \quad 1 \le i < j \le n$

Every braid v_i belongs to the free group $F_i = \left\langle A_{1,i}, A_{2,i}, \cdots, A_{i-1,i} \right\rangle$

So now, deciding whether $v_i \equiv 1$ can be done in the free group $\boldsymbol{F_i}$.

Based on this, appropriated wild and tame logarithmic signatures can be constructed.

Problem: Can it be applied to develop a cryptosystem like MST$_1$ above ?

V.4. Further directions

- The minimal length or Shortest Word Problem ;

- The cycling problem [V. Gebhardt and J. Gonzales-Meneses, 2007], [Gonzales-Meneses, 2011];

- Schemes based on different non-commutative groups [A. Mahalanobis, 2005], [Rosenberger, 2011];

- Schemes based on different associative and non-associative structures [Kalka, 2012];

- The Markov walk situation [Maffre, 2006];

- Combinatorial group theory and cryptography [V. Shpilrain and G. Zapata, 2004], [Kumar, 2011];

- Residual properties of groups in cryptography ? [D. Grigoriev and I Ponamarenko, 2005]

- On which type of groups can we define what kind of difficult problems to build cryptosystems ?

Recapitulative table:

Schemes	Problems	Attacks
AAG (1999)	Multiple CSP	S.J. Lee & E. Lee (2002)
KLCHKP (2000)	CSP	E. Lee & J.H. Park (2003) D. Hofheinz & R. Steinwandt (2003)
LLL (2003)	CDP	D. Hofheinz & R. Steinwandt (2003) J.H. Cheon & B. Jun (2003) E. Lee & J.H. Park (2003)
SDG (2006)	CP, REP Root problem	A. Groch, D. Hofheinz & R. Steinwandt (2006), B. Tsaban (2006)
P. Dehornoy (2006) ?	Shifted CSP	Longrid & Ushakov
Glez Vasco (2003) ?	Markov Chains & Knot theory	?
M.I. Glez Vasco & *al*. (2003) ?	Pure braid (Combing braids)	?
V. Shpilrain & al. 2004 J.C. Birget and *al*. ?	CGT & cryptography	?
….	….	….

Thanks for your kind attention!