

---

# Use of Signed Permutations in Cryptography

*Original research  
papers*

---

## Abstract

In this paper we consider cryptographic applications of the arithmetic on the hyperoctahedral group. On an appropriate subgroup of the latter, we particularly propose to construct public key cryptosystems based on the discrete logarithm. The fact that the group of signed permutations has rich properties provides fast and easy implementation and makes these systems resistant to attacks like the Pohlig-Hellman algorithm. The only negative point is that storing and transmitting permutations need large memory. Using together the hyperoctahedral enumeration system and what is called subexceedant functions, we define a one-to-one correspondence between natural numbers and signed permutations with which we label the message units.

*Keywords:* Signed permutation; public key system; discrete logarithm problem; hyperoctahedral enumeration system, subexceedant function

2010 Mathematics Subject Classification: 11T71; 68R15; 94A60; 97F60

## 1 Introduction

The term cryptography refers to the study of techniques providing information security. Several mathematical objects have been used in this field to label the so-called message units. These objects are often integers. They may also be points or vectors on some curves. For this work, we shall use signed permutations.

Let us denote by:

- $[n]$  the set  $\{1, \dots, n\}$ ,
- $[\pm n]$  the set  $\{-n, \dots, -1, 1, \dots, n\}$ ,
- $\mathcal{S}_n$  the symmetric group of degree  $n$ .

**Definition 1.1.** A bijection  $\pi : [\pm n] \rightarrow [\pm n]$  satisfying  $\pi(-i) = -\pi(i)$  for all  $i \in [\pm n]$  is called "signed permutation".

---

---

We can also write a signed permutation  $\pi$  in the form

$$\pi = \begin{pmatrix} 1 & 2 & \dots & n \\ \varepsilon_1\sigma_1 & \varepsilon_2\sigma_2 & \dots & \varepsilon_n\sigma_n \end{pmatrix} \text{ with } \sigma \in S_n \text{ and } \varepsilon_i \in \{\pm 1\}.$$

Under the ordinary composition of mappings, all signed permutations of the elements of  $[n]$  form a group  $\mathcal{B}_n$  called hyperoctahedral group of rank  $n$ . We write  $\pi^k = \underbrace{\pi \circ \dots \circ \pi}_{k\text{-times}}$  for an integer  $k$  and

$\pi \in \mathcal{B}_n$  and when we multiply permutations, the leftmost permutation acts first. For example,

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & -3 & 4 & 2 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & -2 & 4 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & -4 & 1 & -2 \end{pmatrix}.$$

Before the 1970's, to cipher or decipher a message, two users of cryptographic system must safely exchange information (the private key) which is not known by anyone else. New keys could be periodically distributed so as to keep the enemy guessing. In 1976, W. Diffie and M. Hellman [Diffie and Hellman (1976)] discovered an entirely different type of cryptosystem for which all of the necessary information to send an enciphered message is publicly available without enabling anyone to read the secret message. With this kind of system called public key, it is possible for two parties to initiate secret communications without exchanging any preliminary information or ever having had any prior contact.

The security of public key cryptosystems is based on the hardness of some mathematical problems. As a public key cryptosystem consists of a private key (the deciphering key) that is kept secret and a public key (the enciphering key) which is accessible to the public, then the straightforward way to break the system is to draw the private key from the public key. Therefore, the required computation cost is equivalent to solving these difficult mathematical problems. For instance for the two particularly important examples of public key cryptosystems : RSA and Diffie-Hellman, both are connected with fundamental questions in number theory which are : difficulty of factoring a large composite integer whose prime factors are not known in advance and intractability of discrete logarithm in the multiplicative group of a large finite field, respectively.

**Definition 1.2.** The discrete logarithm problem in the finite group  $G$  to the base  $g \in G$  is the problem : given  $y \in G$ , of finding an integer  $x$  such that  $g^x = y$ , provided that such an integer exists (in other words, provided that  $y$  is in the subgroup generated by  $g$ ).

If we really want our random element  $y$  of  $G$  to have a discrete logarithm,  $g$  must be a generator of  $G$ .

**Example 1.1.** Let  $G = (\mathbb{Z}/19\mathbb{Z})^*$  be the multiplicative group of integers modulo 19. The successive powers of 2 reduced mod 19 are : 2, 4, 8, 16, 13, 7, 14, 9, 18, 17, 15, 11, 3, 6, 12, 5, 10, 1. Let  $g$  be the generator 2, then the discrete logarithm of 9 to the base 2 is 8.

In many ways, the group of signed permutations is analogous to the multiplicative group of a finite field. For this purpose, cryptosystems based on the latter can be translated to systems using the hyperoctahedral group.

This work addresses public key cryptosystems based on the group of signed permutations and related to the discrete logarithm problem. We shall illustrate this in section 3 by an hyperoctahedral group analogue for the Diffie-Hellman key exchange system then by describing two hyperoctahedral group public key cryptosystems for transmitting information. We shall also explain why there is no subexponential time algorithms to break the proposed systems. Before introducing the cryptosystems themselves, we give in section 2 a method to define a bijection between integers and signed permutations. This time, we first convert the natural number in hyperoctahedral system and then we use the result to compute the corresponding signed permutation by means of subexceedant function.

---

## 2 Integer representation of signed permutations

### 2.1 Converting from one base to another

**Definition 2.1.** Hyperoctahedral number system is a system that expresses all natural number  $n$  of  $\mathbb{N}$  in the form :

$$n = \sum_{i=0}^{k(n)} d_i \cdot B_i, \text{ where } k(n) \in \mathbb{N}, d_i \in \{0, 1, 2, \dots, 2i + 1\} \text{ and } B_i = 2^i i!. \quad (2.1)$$

This definition is motivated by the fact that  $B_i$  is the cardinal of the hyperoctahedral group  $\mathcal{B}_i$ .

To denote the non-negative integer  $n$  of the equation (2.1) in the hyperoctahedral system, we use by convention the  $(k + 1)$ -digits representation

$$d_k d_{k-1} \cdots d_2 d_1 d_0.$$

**Theorem 2.1.** *Every positive integer has an unique representation in the hyperoctahedral system.*

Before giving the proof of this theorem, these are some properties of the hyperoctahedral system.

**Lemma 2.2.** *If  $n = d_k \cdots d_1 d_0$  is a number in hyperoctahedral system, then*

$$0 \leq n \leq B_{k+1} - 1$$

*Proof.* Since  $0 \leq d_i \leq 2i + 1$ , we have

$$0 \leq \sum_{i=0}^k d_i B_i \leq \sum_{i=0}^k (2i + 1) B_i.$$

Recall that  $B_i = 2^i i!$ ,

$$\begin{aligned} (2i + 1) B_i &= (2(i + 1 - 1) + 1) B_i \\ &= (2(i + 1) - 1) B_i \\ &= 2(i + 1) B_i - B_i \\ &= B_{i+1} - B_i. \end{aligned}$$

Therefore,

$$\sum_{i=0}^k (2i + 1) B_i = \sum_{i=0}^k (B_{i+1} - B_i) = B_{k+1} - B_0 = B_{k+1} - 1.$$

□

**Lemma 2.3.** *Let  $n = d_k \cdots d_1 d_0$  be a number in hyperoctahedral system, then*

$$d_k B_k \leq n < (d_k + 1) B_k.$$

*Proof.* Let  $m = d_{k-1} \cdots d_1 d_0$ . From lemma 2.2,  $0 \leq m < B_k$ . We also have  $n = d_k B_k + m$ , hence

$$d_k B_k \leq n < B_k + d_k B_k = (1 + d_k) B_k.$$

□

Now, we can proceed to the demonstration of theorem 2.1 which states the one-to-oneness between non negative integers and hyperoctahedral base numbers.

---

*Proof.* Let  $a_n \cdots a_1 a_0$  with  $a_n \neq 0$  and  $b_m \cdots b_1 b_0, b_m \neq 0$  be two representations of a positive integer  $N$  in hyperoctahedral system. First,  $b_m \geq 1$  and  $a_n \geq 1$  imply

$$B_m \leq b_m B_m \leq b_m \cdots b_1 b_0 \text{ and } B_n \leq a_n \cdots a_1 a_0.$$

The relation  $n = m$  is immediate because if  $n < m$  then

$$B_m \geq B_{n+1} > a_n \cdots a_1 a_0$$

by lemma 2.2 so

$$b_m \cdots b_1 b_0 > a_n \cdots a_1 a_0 .$$

Similarly, if  $m < n$  then

$$a_n \cdots a_1 a_0 > b_m \cdots b_1 b_0 .$$

We get a contradiction. Consequently  $n = m$  and  $a_i = b_i$  for all  $i \in \{0, 1, \dots, n\}$  by induction and by the unicity of the expression  $N = a_n B_n + r_n$  with  $r_n = a_{n-1} \cdots a_1 a_0 < B_n$ .  $\square$

To express a positive integer  $n$  in the hyperoctahedral system, one proceeds with the following manner. Start by dividing  $n$  by 2 and let  $d_0$  be the rest  $r_0$  of the expression

$$n = r_0 + (2)q_0 .$$

Divide  $q_0$  by 4, and let  $d_1$  be the rest  $r_1$  of the expression

$$q_0 = r_1 + (4)q_1 .$$

Continue the procedure by dividing  $q_{i-1}$  by  $2(i+1)$  and taking  $d_i := r_i$  of the expression

$$q_{i-1} = r_i + 2(i+1)q_i$$

until  $q_l = 0$  for some  $l \in \mathbb{N}$ . In this way, we obtain  $n = d_l : d_{l-1} : \cdots : d_1 : d_0$  and we also have

$$n = d_0 + 2(d_1 + 4 \cdot (d_2 + 2(3) \cdot (d_3 + \cdots))) .$$

Now let  $n = d_{k-1} : d_{k-2} : \cdots : d_1 : d_0$  be a number in hyperoctahedral system. By definition 2.1, one way to convert  $n$  to the usual decimal system is to calculate

$$d_{k-1} 2^{k-1} (k-1)! + \cdots + d_1 \cdot 2 + d_0 .$$

In practice, one can use this algorithm :

---

Input : An integer  $d_{k-1} : d_{k-2} : \cdots : d_1 : d_0$  in hyperoctahedral system.  
Output : An integer  $d$  in decimal system.

---

1. initiate the value of  $d$  :  $d \leftarrow d_{k-1}$
  2. for  $i$  from  $k-1$  to 1 do :  $d \leftarrow d \cdot 2 \cdot i + d_{i-1}$
  3. return  $d$
- 

## 2.2 A bijection between the subexceedant functions and permutations

**Definition 2.2.** A subexceedant function  $f$  on  $[n]$  is a map  $f : [n] \rightarrow [n]$  such that

$$1 \leq f(i) \leq i \text{ for all } i \in [n] .$$

We will denote by  $\mathcal{F}_n$  the set of all subexceedant functions on  $[n]$ , and we will represent a subexceedant function  $f$  over  $[n]$  by the word  $f(1)f(2)\cdots f(n)$ .

**Example 2.4.** These are the sets  $\mathcal{F}_n$  for  $n = 1, 2, 3$  :

$$\begin{aligned}\mathcal{F}_1 &= \{1\} \\ \mathcal{F}_2 &= \{11, 12\} \\ \mathcal{F}_3 &= \{111, 112, 113, 121, 122, 123\}.\end{aligned}$$

It is easy to verify that  $\text{card } \mathcal{F}_n = n!$ , since from each subexceedant  $f$  over  $[n-1]$ , one can obtain  $n$  distinct subexceedant functions over  $[n]$  by adding any integer  $i \in [n]$  at the end of the word representing  $f$ .

We will give a bijection between  $\mathcal{S}_n$  and  $\mathcal{F}_n$ . Let be the map

$$\begin{aligned}\phi : \mathcal{F}_n &\longrightarrow \mathcal{S}_n \\ f &\longmapsto \sigma_f = (1 f(1))(2 f(2)) \cdots (n f(n))\end{aligned}$$

Notice that there is an abuse of notation in the definition of  $\sigma_f$ . Indeed, if  $f(i) = i$ , then the cycle  $(i f(i)) = (i)$  does not really denote a transposition but simply the identity permutation.

**Lemma 2.5.** The map  $\phi$  is a bijection from  $\mathcal{F}_n$  onto  $\mathcal{S}_n$ .

*Proof.* Since  $\mathcal{S}_n$  and  $\mathcal{F}_n$  both have cardinality  $n!$ , it suffices to prove that  $f$  is injective. Let  $f$  and  $g$  be two subexceedant functions on  $[n]$ . Assume that  $\phi(f) = \phi(g)$  i.e.  $\sigma_f = \sigma_g$ . So we have :

$$(1 f(1))(2 f(2)) \cdots (n f(n)) = (1 g(1))(2 g(2)) \cdots (n g(n)). \quad (2.2)$$

Since  $\sigma_f = \sigma_g$ , then in particular  $\sigma_f(n) = \sigma_g(n)$ . By definition  $\sigma_f(n) = f(n)$  and  $\sigma_g(n) = g(n)$ , so  $f(n) = g(n)$ . Let us multiply both members of equation (2.2) on the right by the permutation  $(n f(n)) = (n g(n))$ , we obtain :

$$(1 f(1))(2 f(2)) \cdots (n-1 f(n-1)) = (1 g(1))(2 g(2)) \cdots (n-1 g(n-1)).$$

Now, if we apply the same process to this equation, we obtain  $f(n-1) = g(n-1)$ . By iterating, we conclude that  $f(i) = g(i)$  for all integers  $i \in [n]$  and then  $f = g$ .  $\square$

Let  $\sigma$  be a permutation of the symmetric group  $\mathcal{S}_n$  and  $f$  be the inverse image of  $\sigma$  by  $\phi$ . Then  $f$  can be constructed as below :

1. Set  $f(n) = \sigma(n)$ .
2. Multiply  $\sigma$  on the right by  $(n \sigma(n))$  (this operation consists in exchanging the image of  $n$  and the image of  $\sigma^{-1}(n)$ ), we obtain a new permutation  $\sigma_1$  having  $n$  as a fixed point. Thus  $\sigma_1$  can be considered as a permutation of  $\mathcal{S}_{n-1}$ . Then set  $f(n-1) = \sigma_1(n-1)$ .
3. In order to obtain  $f(n-2)$ , apply now the same process to the permutation  $\sigma_1$  by multiplying  $\sigma_1$  by  $(n-1 \sigma_1(n-1))$ . Iteration determines  $f(i)$  for all integers  $i$  of  $[n]$ .

## 2.3 Mapping hyperoctahedral base numbers to signed permutations

Let  $d_{n-1} d_{n-2} \cdots d_1 d_0$  be a  $n$ -digits number in hyperoctahedral system. That is  $d_i \in \{0, 1, 2, \dots, 2i+1\}$  for  $i = 0, \dots, n-1$ .

Writing  $d_i = 2q_i + r_i$  where  $r_i \in \{0, 1\}$  and  $q_i \in \{0, \dots, i\}$ , gives

- the subexceedant function  $f = f(1) \cdots f(n)$  with  $f(i) = 1 + q_{i-1}$ ,  $i = 1, \dots, n$
- and the sequence  $(\varepsilon_1, \dots, \varepsilon_n)$  with  $\varepsilon_i = (-1)^{r_{i-1}}$  for  $i = 1, \dots, n$ .

---

So to each integer  $d_{n-1} d_{n-2} \cdots d_1 d_0$ , we associate the signed permutation

$$\begin{pmatrix} 1 & 2 & \cdots & n \\ \varepsilon_1 \sigma_1 & \varepsilon_2 \sigma_2 & \cdots & \varepsilon_n \sigma_n \end{pmatrix}$$

where  $\sigma = \sigma_1 \cdots \sigma_n$  is the permutation associated to the subexceedant function  $f$  by the map

$$\begin{aligned} \phi : \mathcal{F}_n &\longrightarrow \mathcal{S}_n \\ f &\longmapsto \sigma_f = (1f(1))(2f(2)) \cdots (nf(n)) \end{aligned}$$

Now, let  $\pi = \begin{pmatrix} 1 & 2 & \cdots & n \\ \varepsilon_1 \sigma_1 & \varepsilon_2 \sigma_2 & \cdots & \varepsilon_n \sigma_n \end{pmatrix}$  be a signed permutation. As  $\phi$  is a bijection, from  $\pi$  we have

$$f = f(1) \cdots f(n) = \phi^{-1}(\sigma) \text{ and } r_{i-1} = \begin{cases} 0 & \text{if } \varepsilon_i = 1 \\ 1 & \text{if } \varepsilon_i = -1 \end{cases} \text{ for } i = 1, \dots, n.$$

The digits  $d_i = 2(f(i+1)-1)+r_i$ ,  $i = 0, \dots, n-1$  form the hyperoctahedral number  $d_{n-1} d_{n-2} \cdots d_1 d_0$ . It is easy to verify that  $d_i \in \{0, \dots, 2i+1\}$ . We have

$$\begin{aligned} 1 \leq f(i) \leq i & \quad \text{for } i = 1, \dots, n \\ 0 \leq f(i+1) \leq i+1 & \quad \text{for } i = 0, \dots, n-1 \\ 0 \leq 2(f(i+1)-1) \leq 2i & \quad \text{for } i = 0, \dots, n-1 \\ 0 \leq d_i \leq 2i+1 & \quad \text{because } 0 \leq r_i \leq 1. \end{aligned}$$

### 3 Some cryptosystems based on hyperoctahedral group

We suppose that we are using message units with signed permutations as equivalents (according to section 2) in some publicly known large hyperoctahedral group  $\mathcal{B}_n$ .

#### 3.1 Analog of the Diffie-Helman key exchange

The Diffie-Hellman key exchange [Diffie and Hellman (1976)] which was the first public key cryptosystem originally used the multiplicative group of a finite field. It can be adapted for signed permutations group as follow.

Suppose that two users Alice and Bob want to agree upon a secret key, a random element of  $\mathcal{B}_n$ , which they will use to encrypt their subsequent messages to one another. They first choose a large integer  $n$  for the hyperoctahedral group  $\mathcal{B}_n$  and select a signed permutation  $\beta \in \mathcal{B}_n$  to serve as their "base" and make them public. Alice selects a random integer  $0 < a < B_n$ , which she keeps secret, and computes  $\beta^a \in \mathcal{B}_n$  which she makes public. Bob does the same. He selects a random integer  $0 < b < B_n$ , and transmits  $\beta^b$  to Alice over a public channel. Alice and Bob agree on the secret key  $\beta^{ab}$  by computing  $(\beta^b)^a$  and  $(\beta^a)^b$  respectively.

#### 3.2 Analog of the ElGamal system

Suppose user "A" requires sending a message  $\mu$  to user "B". As in the key exchange system above, we start with a fixed publicly known hyperoctahedral group  $\mathcal{B}_n$  for a large integer  $n$  and a signed permutation  $\beta \in \mathcal{B}_n$  (preferably, but not necessarily, a generator) as base. Each user selects a random integer  $0 < u < |\beta|$  (instead of  $u$  we take  $a$  for user "A" and  $b$  for user "B") which is kept secret, and computes  $\beta^u$  that he publishes as a public key. With the public key  $\beta^b$ , user "A" encrypt the message  $\mu$  and sends the pair of signed permutations  $(m_1 = \beta^a, m_2 = \mu \cdot (\beta^b)^a)$  to user "B". To recover  $\mu$ , B computes  $(m_1^b)^{-1} = ((\beta^a)^b)^{-1}$  and multiplies  $m_2$  on the right by the result.

### 3.3 Analog of the Massey-Omura system for message transmission.

We assume the same setup as in the previous subsection. Suppose that Alice wants to send Bob a message  $\mu$ . She chooses a random integer  $c$  satisfying  $0 < c < B_n$  and  $\text{g.c.d.}(c, B_n) = 1$ , and transmits  $\mu^c$  to Bob. Next, Bob chooses a random integer  $d$  with the same properties, and transmits  $(\mu^c)^d$  to Alice. Then Alice transmits  $(\mu^{cd})^{c'} = \mu^d$  back to Bob, where  $c'c \equiv 1 \pmod{B_n}$ . Finally, Bob computes  $(\mu^d)^{d'}$  where  $d'd \equiv 1 \pmod{B_n}$ .

### 3.4 Security

For the Diffie-Hellman's key exchange above, a third party knows only the elements  $\beta$ ,  $\beta^a$  and  $\beta^b$  of  $\mathcal{B}_n$  which are public knowledge. Obtaining  $\beta^{ab}$  knowing only  $\beta^a$  and  $\beta^b$  is as hard as taking the discrete logarithm  $a$  from  $\beta$  and  $\beta^a$  (or  $b$  knowing  $\beta$  and  $\beta^b$ ). Then an unauthorized third party must solve the discrete logarithm problem to the base  $\beta \in \mathcal{B}_n$  to determine the key.

Both ElGamal's [ElGamal (1985)] and Massey-Omura's [Wah and Wang (1984)] cryptosystems are essentially variants of Diffie-Hellman's key exchange system. Therefore, breaking either of the systems above requires the solution of the discrete logarithm problem for hyperoctahedral group.

**Definition 3.1.** The hyperoctahedral discrete logarithm problem to the base  $\beta \in \mathcal{B}_n$  is the problem, given  $\pi \in \mathcal{B}_n$ , of finding an integer  $x$  such that  $\pi = \beta^x$  if such  $x$  exists.

Many improvements are made for solving the discrete logarithm problem in finite group. In hyperoctahedral group based cryptosystems of the sort discussed above, one does not work with the entire group  $\mathcal{B}_n$ , but rather with cyclic subgroups: the group  $\langle \beta \rangle$  in the Diffie-Hellman system and ElGamal system and the group  $\langle \mu \rangle$  in the Massey-Omura system where we designate by

- $\langle \pi \rangle = \{\pi^i \mid i = 1, \dots, |\pi|\}$  the cyclic subgroup of  $\mathcal{B}_n$  generated by  $\pi \in \mathcal{B}_n$ ,
- $|\pi|$  the order of the signed permutation  $\pi$ .

In other words, one works in the subgroup generated by the base of the discrete logarithm in each proposed system.

**Choice of a suitable base** The order of the cyclic subgroup of  $\mathcal{B}_n$  has an important role to avoid an easy solution of the discrete logarithm. Recall that we can also represent a signed permutation  $\pi$  in the form

$$\pi = \begin{pmatrix} 1 & 2 & \dots & n \\ \varepsilon_1\sigma(1) & \varepsilon_2\sigma(2) & \dots & \varepsilon_n\sigma(n) \end{pmatrix} \text{ with } \sigma \in S_n \text{ and } \varepsilon_i \in \{\pm 1\}.$$

As stated in the work of Victor Reiner [Reiner (1993)], a signed permutation decomposes uniquely into a product of commuting cycles just as permutations do.

**Example 3.1.** The disjoint cycle form of  $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 3 & 6 & -2 & 7 & -5 & -1 & 4 \end{pmatrix}$  is

$$\begin{pmatrix} 1 & 3 & 2 & 6 \\ 3 & -2 & 6 & -1 \end{pmatrix} \begin{pmatrix} 4 & 7 \\ 7 & 4 \end{pmatrix} \begin{pmatrix} 5 \\ -5 \end{pmatrix}.$$

**Definition 3.2.** The order of a signed permutation  $\pi$  is the smallest positive integer  $m$  such that  $\pi^m = \iota$  where  $\iota$  denotes the identity permutation.

An  $\ell$ -cycle  $C = \begin{pmatrix} i_1 & i_2 & \dots & i_{\ell-1} & i_\ell \\ \varepsilon_1 i_2 & \varepsilon_2 i_3 & \dots & \varepsilon_{\ell-1} i_\ell & \varepsilon_\ell i_1 \end{pmatrix}$  where  $\varepsilon_i \in \{\pm 1\}$  has order  $\ell$ . The proof of the following theorem can be found in [Pohlig and Hellman (2002)].

---

**Theorem 3.2.** *The order of a permutation written in disjoint cycle form is the least common multiple of the lengths of the cycles.*

Let us now assume that we have  $\beta \in \mathcal{B}_n$  as base of the discrete logarithm. The order of  $\langle \beta \rangle$  can be very large for a large value of  $n$ , since the disjoint cycles of  $\beta$  can be selected in such a way that their least common multiple be very large. Therefore brute force search is inefficient to solve the discrete logarithm problem to the base  $\beta \in \mathcal{B}_n$ .  $\langle \beta \rangle$  is a cyclic group of finite order so it is commutative. Thus, in order to avoid an easy solution to the discrete logarithm problem using the techniques that apply to any finite abelian group (which take approximately  $\sqrt{p}$  operations, where  $p$  is the largest prime dividing the order of the group), it is important for the order of the commutative group  $\langle \beta \rangle$  to be non smooth, that is, divisible by a large prime.

**Definition 3.3.** Let  $n$  be a positive real number. we say that  $n$  is smooth if all of the prime factors of  $n$  are small.

The method of Pohlig-Hellman [Pohlig and Hellman (1978)] can efficiently computes the discrete logarithm in a group  $G$  if its order is  $B$ -smooth for a reasonably small  $B$ .

**Definition 3.4.** Let  $B$  be a positive real number. An integer is said to be  $B$ -smooth if it is not divisible by any prime greater than  $B$ .

The following theorem allows to generate the base  $\beta$  so that the order of the subgroup  $\langle \beta \rangle$  of  $\mathcal{B}_n$  have arbitrary smoothness.

**Theorem 3.3.** *Let  $p$  be a prime. For a large integer  $n$ , a cyclic subgroup  $G$  of  $\mathcal{B}_n$  which its order is  $p$ -smooth and is not  $(p-1)$ -smooth can be constructed.*

*Proof.* Let  $n$  be a large integer and  $p \leq n$  a prime. There exists a  $p$ -cycle  $\gamma_p \in \mathcal{B}_n$ . Let  $\gamma_p, C_1, C_2, \dots, C_k \in \mathcal{B}_n$  be disjoint cycles of length respectively  $p, l_1, l_2, \dots, l_k$  with

$$1 \leq l_i \leq p \text{ for } i = 1, \dots, k \text{ and } \sum_{i=1}^k l_i \leq n - p .$$

Let us now consider the signed permutation  $\pi = \gamma_p C_1 C_2 \dots C_k \in \mathcal{B}_n$  of order

$$|\pi| = lcm(p, l_1, l_2, \dots, l_k) .$$

Let  $q$  be a prime such that  $q | lcm(p, l_1, l_2, \dots, l_k)$ . We show that  $q \leq p$ . Let us suppose that  $q > p$ . We obtain  $l_i < p < q$  which implies  $q \nmid |\pi|$  so  $q \leq p$ . We have just shown that every prime  $q$  dividing the order of  $\pi$  is smaller than  $p$ , that is,  $|\pi|$  is  $p$ -smooth. The order  $|\langle \pi \rangle| = |\pi|$  of the cyclic subgroup  $\langle \pi \rangle$  of  $\mathcal{B}_n$  generated by  $\pi$  is  $p$ -smooth but it is not  $(p-1)$ -smooth because  $p \mid |\langle \pi \rangle|$  and  $p > p-1$ .  $\square$

## 4 Conclusions

Theorem 3.3 provides high flexibility in selecting a subgroup of hyperoctahedral group  $\mathcal{B}_n$  on which cryptosystem resists to attacks by Silver-Pohlig-Hellman's algorithm.

The cryptosystems that we have proposed are easy to implement by applying optimized method for exponentiation. Moreover, the multiplication on hyperoctahedral group  $\mathcal{B}_n$  which is the composition of mappings can be performed in time  $\mathcal{O}(n)$ . However, as we must work in a very large hyperoctahedral group, the need of large memory from the point of view implementation requires improvements.



---

## References

- Diffie, W. and Hellman, M. E. (1976). New directions in cryptography. IEEE Transactions on Information Theory IT-22, 644-654.
- ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory IT-31, 469-472.
- Pohlig, S.C. and Hellman, M.E. (1978). An improved algorithm for computing logarithms over  $GF(p)$  and its cryptographic significance. IEEE Transactions on Information Theory 24, 106-110.
- Rotman, J.J. (2002). Advanced Modern Algebra. Prentice Hall, 1st edition, New Jersey.
- Reiner, V. (1993). Signed Permutation Statistics and Cycle Type. European Journal of Combinatorics 14, 569-579.
- Wah, P. K. S. and Wang, M. Z. (1984). Realization and application of the Massey-Omura lock. Proc. International Zurich Seminar, pages 175–182.