# COMPLEX: Projet sur les Tests de Primalité

Pour le Mardi, 25 Novembre, 2014

**Mathieu Ville, Arthur Ramolet**

# Table des matières

# 1 Introduction

# 2 Arithmétique dans Zn

Cette première partie du projet propose de développer quelques fonctions simples utiles pour la cryptographie. Celles-ci serviront par la suite mais permettent surtout une première prise en main de gmp, la bibliothèque grand nombres du langage C.
Toutes les fonctions sont vrifies en comparant leurs rsultats avec ceux des fonctions gmp existantes.

## 2.1 My_pgcd

La fonction my_pgcd (Voir listing 1) permet le calcul du pgcd de deux nombres. Celle-ci existe déjá sous le nom mpz_gcd. Pour les raisons évoquées dans le paragraphe précédent nous avons souhaités réaliser notre propre fonction.
La fonction my_pgcd nécessitant l'utilisation de variables temporaires (Allouées dynamiquement par gmp) elle a été implémentée de sorte á éviter l'allocation dans la partie récursive de la fonction (my_pgcdMemSave).

Listing 1 – My_pgcd avec gmp

```
void mpz_max(mpz_t result, mpz_t a,mpz_t b){
     if(mpz_cmp(a,b)>=0)
          mpz_set(result,a);
     else
          mpz_set(result,b);
}

void mpz_min(mpz_t result, mpz_t a,mpz_t b){
     if(mpz_cmp(a,b)<=0)
          mpz_set(result,a);
     else
          mpz_set(result,b);
}

void mpzMy_pgcd(mpz_t result,mpz_t a, mpz_t b){
     mpz_t min;
     mpz_t max;
     mpz_t tmp;
     mpz_init (min);
     mpz_init (max);
     mpz_init (tmp);
     mpzMy_pgcdMemSave(result,a,b,min,max,tmp);
     mpz_clear(min);
     mpz_clear(max);
     mpz_clear(tmp);
}

void mpzMy_pgcdMemSave(mpz_t result,mpz_t a, mpz_t b,mpz_t min,mpz_t max,mpz_t tmp){
     if(mpz_cmp(a,b)==0)
          mpz_set(result,a);
     else{
          mpz_max(max,a,b);
          mpz_min(min,a,b);
          mpz_sub(tmp,max,min);
          mpzMy_pgcdMemSave(result,tmp,min,min,max,tmp);
     }
}
```

## 2.2 My_inverse

La fonction my_inverse permet de calculer l'inverse du modulo d'un entier (voir listing 2). La fonction a été implémenté avec l'algorithme d'Euclide étendu.
La fonction gmp existante s'appelle : my_invert.

Listing 2 – my_inverse avec gmp

```
void mpzMy_inverse(mpz_t result, mpz_t a, mpz_t n)
{
     mpz_t t, nt, r, nr, q, tmp,tmpa, tmpn;

     mpz_init (t);
     mpz_init (nt);
     mpz_init (r);
     mpz_init (nr);
     mpz_init (q);
     mpz_init (tmp);
     mpz_init (tmpa);
     mpz_init (tmpn);

     if (mpz_sgn(n) == -1)
          mpz_neg(tmpn,n);
     if (mpz_sgn(a) == -1) {
          mpz_neg(tmpa,a);
          mpz_mod(tmpa,tmpa,n);
          mpz_sub(tmpa, n,tmpa);
     }

     mpz_set(tmpn,n);
     mpz_set(tmpa,a);
     mpz_set_str(nt,"1",10);
     mpz_set(r,tmpn);

     mpz_mod(nr,a,tmpn);

     while (mpz_sgn(nr) != 0) {

          mpz_tdiv_q(q,r,nr);

          mpz_set(tmp,nt);
          mpz_mul(nt,q,nt);
          mpz_sub(nt,t,nt);
          mpz_set(t,tmp);

          mpz_set(tmp,nr);
          mpz_mul(nr,q,nr);
          mpz_sub(nr,r,nr);
          mpz_set(r,tmp);
     }

     if (mpz_sgn(t) == -1){
          mpz_add(t,t,tmpn);
          mpz_set(result,t);
     }
```

```
    mpz_set(result,t);

    if (mpz_cmp_d(r,1.0) < 0){ /*plus grand que 1*/
        mpz_set_str(result,"-1",10); /* No inverse */
    }

    mpz_clear(t);
    mpz_clear(nt);
    mpz_clear(r);
    mpz_clear(nr);
    mpz_clear(q);
    mpz_clear(tmp);
    mpz_clear(tmpa);
    mpz_clear(tmpn);
}
```

## 2.3 Expo_mod

La fonction expo_mod (voir listing 3) permet le calcul de lexponentiation rapide modulaire (Suivant le principe de l'algo fast exp).
La fonction gmp existante s'appelle : mpz_powm.

Listing 3 – expo_mod avec gmp

# 3 Test Naïf

## 3.1 first_test

On réalise fonction first_test (voir listing 4) permettant de vérifier de faon naïve la primalité de N.

Listing 4 – first_test

## 3.2 Test de la fonction

On souhaite connaitre le plus grand entier qu'il est possible de tester avec notre algorithme au bout d'une minute. Cette recherche s'effectue avec une boucle et un flag de compilation -O2 afin d'optimiser le code.
Le plus grand nombre que l'on peut trouver de la sorte est : (Manque resultat).

# 4 Nombres de Carmichaël

Listing 5 – is_carmichael

Listing 6 – gen_carmichael

# 5 Test de Fermat

Listing 7 – testFermat

# 6 Test de Rabin et Miller

Listing 8 – testRabinMiller

# 7 Conclusion