cryptographie

par David Fernandez, Sinicha Mijajlovic, David Quin, élèves de 1°S1 (modules) du lycée Louise Michel de Bobigny (93)

enseignant : M. François Gaudel

chercheur : M. Daniel Barsky

27.03.95
Compte-rendu du parrainage
Sujet : codage décodage cryptographie.
J'ai compris le but du sujet qui consiste à coder décoder des informations confidentielles, comme celles données par les banques, ou autres organismes.

Seulement je regrette que cet exposé n'ait pas été plus "clair". En effet, le niveau demandé par certaines démonstrations était, je pense, trop élevé pour des élèves de lycée.
Marseille II.

Dans notre société, on utilise beaucoup de codes que ce soit pour les cartes de crédit, les transmissions de banques à banques, ou pour protéger l'information contre l'espionnage …

Aujourd'hui la solution la plus retenue pour le codage est basée sur l'arithmétique modulaire et sur la difficulté qu'il y a à factoriser un entier en ses facteurs premiers (les méthodes que l'on connaît actuellement ne nous permettent pas de le faire en un temps raisonnable).

Avant de voir le codage et le décodage, nous allons définir la notion de congruence, voir quelques-unes de ses propriétés et étudier l'arithmétique modulo m (c'est-à-dire les différentes opérations utilisant les congruences telles que l'addition, la multiplication, l'opposé, l'inverse, etc …).

notion de « modulo m »

Soit m un entier naturel non nul et différent de 1. On dit que deux entiers positifs x et x' sont congrus modulo m si leurs restes dans la division par m sont égaux :

$x \equiv x' \ [m]$ si m divise $x - x'$ c'est-à-dire si $x - x' = mq$ avec $q \in \mathbb{Z}$.

[NDLR :
Des exemples !

• Deux entiers sont congrus modulo 2 s'ils ont la même parité : 4 et 8 sont congrus modulo 2 puisque 8 – 4 (= 4) est divisible par 2 ; 7 et 153 sont congrus modulo 2 puisque 153 – 7 (= 146) est divisble par 2.

• Deux entiers sont congrus modulo 10 s'ils ont même chiffre des unités : leur différence aura alors 0 comme chiffre des unités et elle sera bien divisible par 10. Le lecteur pourra vérifier que si la différence est divisible par 10, les deux nombres ont forcément même chiffre des unités. On peut donc regrouper les entiers en dix "classes" : ceux qui sont congrus à 0 (qui ont 0 pour chiffre des unités) ou à 1, ou à 2, ou à 3, ou à 4, ou à 5, ou à 6, ou à 7, ou à 8, ou à 9.]

définition de Z/*m*Z

L'ensemble Z/*m*Zest l'ensemble des classes modulo *m* que l'on note : $\bar{a}$, $\bar{b}$, …

$\bar{a}$ est la classe de *a* modulo *m* ; c'est l'ensemble de tous les entiers congrus à *a* modulo *m*. De même la classe $\bar{b}$ est l'ensemble de tous les entiers congrus à *b*.

Deux classes $\bar{a}$ et $\bar{b}$ sont confondues si *a* et *b* sont congrus (modulo *m*).

quelques propriétés

[NDLC : la notation « $m \mid a - b$ » doit se comprendre comme « m divise a – b »]

• $a \equiv a\ [m]$
car $a - a = 0$ et 0 est toujours multiple de *m*.

• $a \equiv b \Rightarrow b \equiv a\ [m]$:
si $m \mid a - b$, alors $m \mid b - a$.

• $a \equiv b$ et $b \equiv c\ [m] \Rightarrow a \equiv$ c $[m]$:
m divise $a - b$ et $b - c$ donc il divise $a - c$ $(= (a - b) + (b - c))$.

addition avec Z/*m*Z

On veut démontrer que :

si $x \equiv x'\ [m]$ et $y \equiv y'\ [m]$

alors $x + y \equiv x' + y'\ [m]$.

$x \equiv x'\ [m] \Leftrightarrow x - x' = m\,q$
$y \equiv y'\ [m] \Leftrightarrow y - y' = m\,q'$

$(x+y)-(x'+y') = m(q+q')$
$\Leftrightarrow x+y \equiv x'+y'\ [m]$

On peut donc définir : $\bar{x} + \bar{y} = \overline{x+y}$.

élément neutre de l'addition avec Z/*m*Z

Il y a un élément neutre pour l'addition :

$$\bar{x} + \bar{0} = \bar{0} + \bar{x} = \bar{x}$$

multiplication avec Z/*m*Z

On veut démontrer que

si $x \equiv x'\ [m]$ et $y \equiv y'\ [m]$

alors $x\,y \equiv x'\,y'\ [m]$.

$x \equiv x'\ [m] \Leftrightarrow x - x' = m\,q$
$y \equiv y'\ [m] \Leftrightarrow y - y' = m\,q'$

$x\,y - x'\,y' = x\,(y-y') + y'\,(x-x')$
$= xmq' + y'mq$
$= m\,(xq'+y'q)$

Donc $x\,y \equiv x'\,y'\ [m]$.

On peut donc définir : $\bar{x} \times \bar{y} = \overline{x \times y}$.

élément neutre de la multiplication avec Z/*m*Z

Il y a un élément neutre pour la multiplication :

$$\bar{x} \times \bar{1} = \bar{1} \times \bar{x} = \bar{x}$$

opposé avec Z/*m*Z

La classe $\overline{-a}$ est évidemment l'opposé de $\bar{a}$.

inverse avec Z/*m*Z

Soient :
• *m* premier
• $\bar{a} \in \{\bar{1}, \bar{2}, \bar{3}, \ldots, \overline{m-1}\}$; *a* est tel que *m* ne divise pas *a*. (Nous avons là toutes les classes non nulles.)
On choisit *a* dans {1, 2, 3, … , *m*–1}.
• un nombre *k* tel que $m \geq k \geq 1$

Parmi les nombres a^1, a^2, a^3, …, a^m il y en a au moins deux qui sont congrus modulo *m* car il y a *m* termes et seulement $m - 1$ classes (ils sont tous premiers avec *m*, donc ne peuvent être congrus à 0 modulo *m*).

Donc il existe k_1 et $k_2 \in \{1, 2, 3, \ldots, m\}$ avec $k_1 \neq k_2$ tels que

$$a^{k_1} \equiv a^{k_2}\ [m].$$

[L'un des deux nombres k_1 et k_2 est le plus grand des deux ; appelons k_1 le plus grand :] $k_1 \geq k_2$. Alors :

$$a^{k_1} - a^{k_2} \equiv 0\ [m] \Leftrightarrow a^{k_2}\,(a^{k_1-k_2} - 1) \equiv 0\ [m]$$

m divise le premier membre ; or a^{k_2} est premier avec m, donc m divise $a^{k_1-k_2} - 1$. D'où :

$$a^{k_1-k_2} \equiv 1\ [m]$$

Il existe donc des puissances strictement positives de a appartenant à la classe de 1. Soit g la plus petite d'entre elles : $a^g \equiv 1\ [m]$. g n'est égal à 1 que si a vaut 1 (donc $a = 1$ est alors son propre inverse) ; sinon, $g - 1 \neq 0$ et $a^{g-1}\, a \equiv 1\ [m]$ …

Toute classe **non nulle** $\bar{a}$ admet donc un inverse : $\bar{a}^{g-1}$.

théorèmes de Fermat et d'Euler

théorème de Fermat :
si m est premier, alors $a^{m-1} \equiv 1[m]$

démonstration :

On se place dans $\mathbb{Z}/m\mathbb{Z}$: soient m premier, $\bar{a} \in \mathbb{Z}/m\mathbb{Z}$ et $\bar{a} \neq \bar{0}$.

Formons les produits :

$$\bar{a} \times \bar{1},\ \bar{a} \times \bar{2},\ \bar{a} \times \bar{3},\ \ldots,\ \bar{a} \times \overline{m-1}.$$

Aucun n'est nul car parmi les produits $a \times 1$, $a \times 2$, $a \times 3$, …, $a \times (m-1)$, aucun n'est congru à 0 modulo m ; or ils sont tous différents car :

$$\bar{a} \times \bar{i} = \bar{a} \times \bar{j} \Leftrightarrow \bar{a} \times \bar{i} - \bar{a} \times \bar{j} = \bar{0}$$
$$\Leftrightarrow \bar{a}\,(\bar{i} - \bar{j}) = \bar{0}$$

ce qui est impossible puisqu'il faudrait que $\bar{a} = \bar{0}$ ou $\bar{i} = \bar{j}$.
[NDLR : puisque $\bar{a}$ n'est pas nulle, elle est inversible, et on peut multiplier les deux membres de $\bar{a}\,(\bar{i} - \bar{j}) = \bar{0}$ par cet inverse, ce qui prouvera immédiatement que c'est $\bar{i} - \bar{j}$ qui est nulle.]

Il y a donc $m-1$ termes différents et $m-1$ valeurs possibles ($\bar{1}$, $\bar{2}$, $\bar{3}$, …, $\overline{m-1}$) donc ces produits reproduisent tous les termes de $\mathbb{Z}/m\mathbb{Z}$, donc :

$$\bar{a} \times \bar{1} \times \bar{a} \times \bar{2} \times \bar{a} \times \bar{3} \times \ldots \times \bar{a} \times \overline{m-1}$$
$$= \bar{1} \times \bar{2} \times \bar{3} \times \ \ldots \times \overline{m-1} = \overline{(m-1)!}$$

Or :

$$a \times 1 \times a \times 2 \times a \times 3 \times \ldots \times a \times (m-1)$$
$$= a^{m-1}(m-1)!$$

Donc :

$$\bar{a}^{m-1}\,\overline{(m-1)!} = \overline{(m-1)!}$$
$$\bar{a}^{m-1} = 1$$

En effet $(m-1)!$ ne peut être congru à 0 modulo m car sinon il serait divisible par m. Or quand un nombre premier divise un produit de facteurs, il divise au moins l'un des facteurs ; et ici, tous les facteurs sont plus petits que m. $(m-1)!$ n'étant pas congru à 0 modulo m, il est donc inversible d'après le paragraphe précédent. On peut donc simplifier par $(m-1)!$ et on a bien :

$$\bar{a}^{m-1} = 1, \text{ soit :}$$
$$a^{m-1} \equiv 1\ [m].$$

indicatrice d'Euler :

Lorsque m n'est pas premier, on définit un nombre entier appelé indicatrice d'Euler de m $\varphi(m)$. Dans le cas où m est le produit de deux nombres premiers, p et q, on a :

$$\varphi(m) = (p-1)(q-1)$$

Nous admettons le **théorème d'Euler :**

Si a et m sont premiers entre eux,
alors $a^{\varphi(m)} \equiv 1\ [m]$

Exemple : $m = 6$; $\varphi(m) = (3-1)(2-1) = 2$.

Les nombres ($a =$) 1 ou ($a =$) 5 sont premiers avec ($m =$) 6, et leurs carrés (1 ou 25) sont bien congrus à 1 modulo ($m =$) 6.

codage & décodage

On sait donc que :

• soit r un nombre entier positif et $\varphi(r)$ son indicatrice d'Euler ; si a et r premiers entre eux, alors $a^{\varphi(r)} \equiv 1\ [r]$

• De plus, si a et b sont premiers entre eux, il existe k et j dans Z tels que $ka + jb = 1$. C'est l'identité de Bezout.

méthode qui permet de trouver des termes pour l'identité de Bezout :

On utilise l'*algorithme d'Euclide*, qui permet de trouver le plus grand diviseur commun à deux entiers … exemple pour $a = 173$ et $b = 32$:

• On effectue la division en nombres entiers de 173 par 32 … $173 = 32 \times 5 + 13$. Tout diviseur commun à 173 et 32 est aussi diviseur de 13 ; tout diviseur commun à 32 et 13 est aussi diviseur de 173 ; le plus grand diviseur commun à 173 et 32 est aussi le plus grand diviseur commun à 32 et 13 … On conserve le diviseur 32 et le reste 13, et on recommence :
• division de 32 par 13 … $32 = 13 \times 2 + 6$. On conserve le diviseur et le reste, et on recommence :
• division de 13 par 6 … $13 = 6 \times 2 + 1$. Le plus grand diviseur commun à 173 et 32 est donc 1 : 173 et 32 sont premiers entre eux (on aurait pu le savoir plus vite puisque 32 est une puissance de 2).

On peut résumer les étapes de l'algorithme d'Euclide dans un tableau :

	← diviseurs et restes →			
173	32	13	6	1
	5	2	2	
	← quotients →			

Si on “remonte” les calculs effectués :
$1 = 13 - 2 \times 6 = 13 - 2\,(32 - 2 \times 13)$
$= 5 \times 13 - 2 \times 32 = 5\,(173 - 5 \times 32) - 2 \times 32$
$= 5 \times 173 - 27 \times 32$

Donc $1 = 5 \times 173 - 27 \times 32$: on a trouvé une identité de Bezout avec $k = 5$ et $j = -27$.

On peut choisir $k > 0$ (comme on vient de le trouver) ou $j > 0$:

$$\begin{array}{l} 1 = \ \ 5 \times 173 \ - \ 27 \times 32 \\ \underline{0 = -173 \times 32 + 173 \times 32} \\ 1 = -27 \times 173 + 146 \times 32 \end{array}$$

On trouve ici une identité de Bezout avec $k = -27$ et $j = 146$.

Maintenant on peut **coder** et **décoder**.

Soit $r > 0$ et s premier avec $\varphi(r)$.

On publie *r* et *s*. $\varphi(r)$ est secret et difficile à calculer. En effet, $r = pq$ avec p et q premiers et très grands (au moins 100 chiffres).

$$\varphi(r) = (p-1)(q-1).$$

Connaissant s et $\varphi(r)$, celui qui décode peut par l'algorithme d'Euclide trouver $t > 0$, tel que $st \equiv 1\ [\varphi(r)]$.

En effet, on résout $st + k\varphi(r) = 1$. Comme s et $\varphi(r)$ sont premiers entre eux, on sait qu'on peut trouver $t > 0$ tel que :

$$\begin{array}{l} st = 1 - k\varphi(r) \text{ avec } k < 0 \\ \quad = 1 + k'\varphi(r) \text{ avec } k' > 0. \end{array}$$

[NDLR : st est positif, $\varphi(r)$ est très grand, positif, et pour que $1 - k\varphi(r)$ soit positif, k est forcément négatif]

Codage préliminaire

Le codeur convertit le message en un nombre M (par exemple en remplaçant chaque lettre de l'alphabet par un chiffre ; $a = 0$, $b = 1$, …). Il faut que $1 < M < r$ (on code des groupes de lettres pas trop grands) et que M et r soient premiers entre eux.

Codage effectif

On calcule $E \equiv M^s\ [r]$ et E est le message codé avec $0 < E < r$.

Décodage

Pour décoder, on se sert de t et de $\varphi(r)$. Puisqu'on a $E \equiv M^s\ [r]$, alors on a :

$$E^t \equiv M\ [r].$$

En effet, $E^t \equiv (M^s)^t = M^{st} = M^1 \times M^{k'\varphi(r)} = M^1 \times (M^{\varphi(r)})^{k'} \equiv M \times (1)^{k'} \equiv M\ [r]$. Et on décode ainsi le message. (On est revenu au code préliminaire.)

Exemples traités avec " Maple "

Nous avons traduit en français et adapté les exemples d'une feuille de calcul fournie avec le logiciel mathématique « Maple ».

Tout d'abord, un petit programme effectue le codage préliminaire du texte choisi : à chaque lettre est associé un code, qui va de 1 à 26, et 27 pour un espace. Ce programme se nomme " codage_preliminaire ", et sera appelé en temps utile.

```
codage_preliminaire : =proc(st)
local ll,nn,ss,ii,num ;
num:=table(['a'=1,'b'=2,'c'=3,'d'=4,'e'=5,'f'=6,'g'=7,
'h'=8,'i'=9,'j'=10,'k'=11,'l'=12,'m'=13,'n'=14,'o'=15,
'p'=16,'q'=17,'r'=18,'s'=19,'t'=20,'u'=21,'v'=22,
'w'=23,'x'=24,'y'=25,'z'=26,`` `=27]) :
if not type(st,string) then ERROR(`wrong number
(or type) of arguments`) fi ;
ll : =length(st) ;
if ll= 0 then RETURN(0) fi ;
nn : =1 ;
for ii from 1 to ll do
ss : =num[substring(st,ii..ii)] ;
if(not type(ss,numeric)) then ERROR(`wrong num-
ber (or type) of arguments`) fi ;
nn : =100*nn+ss ;
od ;
nn–10^(2*ll) ;
end :
```

Un autre petit programme effectue maintenant l'opération inverse ; il s'appelle " decodage_final ".

```
decodage_final : =proc(nn)
local ss,mm,ll,ii,ans,a,b,c,d,e,f,g,h,i,j,k,l,m,n,
o,p,q,r,s,t,u,v,w,x,y,z,``,alpha ;
alpha : =table([1=a,2=b,3=c,4=d,5=e,6=f,7=g,8=h,
9=i,10=j,11=k,12=l,13=m,14=n,15=o,16=p,17=q,
18=r,19=s,20=t,21=u,22=v,23=w,24=x,25=y,26=z,
27=` `]) :
mm : =nn ;
if(not type(nn,integer)) then ERROR(`wrong num
ber (or type) of arguments`) fi ;
ll : =floor(trunc(evalf(log10(mm)))/2)+1 ;
ans : =`` ;
for ii from 1 to ll do
mm : =mm/100 ;
ss : =alpha[frac(mm)*100] ;
if(not type(ss,string)) then ERROR(`wrong number
(or type) of arguments`) fi ;
ans : =cat(ss, ans) ;
mm : =trunc(mm)
od ;
ans ;
end :
```

Exemple d'utilisation des ces deux programmes :

```
> codage_preliminaire(`louise michel`) ;
          12152109190527130903080512
> decodage_final(02150209071425) ;
                      bobigny
```

• On va maintenant rechercher un nombre $r = p \times q$ (avec p et q premiers).

On choisit d'abord deux grands nombres premiers à l'aide de la commande « nextprime » qui donne le nombre premier suivant de ….

```
> p:=nextprime(1234567891011121314151617) ;
          p:=1234567891011121314151753

> q:=nextprime(3029282726252423222212019181716
151413121110) ;
q:=3029282726252423222212019181716151413121133

> r:=p*q ;
r:=3739855186625874075938848859494950431867\
          855820960337965524133296149
```

• Calcul de l'indicatrice d'Euler de r :

```
> phi_r : =(p–1)*(q–1) ;
phi_r:=3739855186625874075938845830212224l7\
       9444632374373265238251406023264
```

• Nous sélectionnons s, qui sera publié avec r, et qui doit être premier avec phi(r).

Il suffit de prendre s premier et strictement inférieur à phi(r), tel que le quotient de phi(r) par s ne tombe pas juste : on vérifie ce dernier point en effectuant la division (avec 200 chiffres significatifs, ce qui est beaucoup trop, mais pourquoi se priver ?).

```
> s : =nextprime(314159) ;
                      s:=314161

> evalf(phi_r/s,200) ; .119042630581958743317561\
563345298244512992776772841480586431989434 2\
073013518546223114899685193260780300546535 0\
568657471805857506183135398728677334233084 3\
102740314679415968245581087404229041797040 3\
7102 10^62
```

• Nous devons maintenant trouver t tel que : st soit congru à 1 modulo phi(r).

Pour cela, on résout l'équation : $st + k\text{phi}(r) = 1$, à l'aide de l'algorithme d'Euclide. La commande « igcdex » fait ce travail pour nous : elle renvoie le pgcd, c'est-à-dire 1, et donne à t et k les bonnes valeurs.

Afin d'avoir une valeur de t pas trop grande, on prend son reste modulo phi(r).

```
igcdex(s,phi_r,'t','k') ;
                          1

> t : =t mod phi_r ;
t:=485812975404973631478968740012162135857 5\
          2352200996608227322894888 1
```

Nous disposons maintenant de tous les outils nécessaires à un codage et un décodage.

Exemple de codage :

```
> M : =codage_preliminaire('bonjour') ;
                    M:=2151410152118
```

Codage définitif :
On calcule M^s modulo r (r et s sont dans le public) : on obtient le message codé.

```
> C : =Power(M,s) mod r ;
C:=59640819077783150418119384003801964705395\
          9482305672338685086972246
```

Décodage :
Cette fois, on utilise t, connu seulement du décodeur.

```
> Power(C,t) mod r ;
                    2151410152118

> decodage_final(“) ;
                       bonjour
```

Autres exemples :

```
> M : =codage_preliminaire(`math en jeans`) ;
         M:=13012008270514271005011419

> C : =Power(M,s) mod r ;
C:=18866032853449552279687742481088198417685\
          3450830647791859756353812425
> Power(C,t) mod r ;
          13012008270514271005011419

> decodage_final(“) ;
                  math en jeans
```

(applaudissements nourris dans la salle …)

[NDLC : Lors des transferts de disquettes d'un compatible PC à un Macintosh, on découvre parfois des choses étonnantes :

C : \MSOFFICE \WINWORD \MODELES \NORMAL.DOT- THÉORÈME DE FERMAT- lycee louise michel- lycee louise michel @ÜÕrH—∫@@ÜÕrH—David Ryan QUIN E : \GAMES \PINBALL \DAV.DOC David Ryan QUIN E : \GAMES \PINBALL \DAV.DOC David Ryan QUIN A : \DAV.DOC lycee louise michel C : \WINDOWS \DAV.DOC lycee louise michel A : \DAV.DOC David Ryan QUIN A : \DAV.DOC trani A : \DAV.DOC lycee louise michel$ C : \MSOFFICE \WINWORD \MATHJEAN \DAV.DOC lycee louise

etc … Méfiez-vous, 1984 est passé, mais Big Brother est toujours là, qui veille sur le meilleur des mondes ?]