Université Bordeaux 1 — **MHT812/MHT839 – Calcul Formel**
Mathématiques — Année 2012–2013

Examen, 21 avril 2008, 8h30 – 11h30

Exercice I –

1) a) On pose $\delta := \mathrm{pgcd}(\ell - 1, t)$. Comme ℓ est premier, $(\mathbb{Z}/\ell\mathbb{Z}) \setminus \{0\}$ est un groupe cyclique d'ordre $\ell - 1$, disons engendré par g. Comme 0 n'est pas solution, toute solution a vérifie $a \in (\mathbb{Z}/\ell\mathbb{Z})^*$ et $a^{\ell-1} = 1$; par Bézout, il existe u, v entiers tels que

$$u(\ell - 1) + vt = \delta.$$

On en déduit $a^\delta = (a^t)^v (a^{\ell-1})^u = 1$, qui implique $a^t = 1$ puisque $\delta \mid t$. Finalement, $a^t = 1$ est équivalent à $a^\delta = 1$. On conclut en invoquant un résultat général sur les groupes cycliques : si $d \mid n$, l'équation $x^d = 1$ a exactement d solution dans un groupe cyclique d'ordre n.

On peut aussi raisonner directement : cette équation de degré δ a au plus δ solution (dans le corps commutatif $\mathbb{Z}/\ell\mathbb{Z}$), et les $g^{x(\ell-1)/\delta}$, $0 \leqslant x < \delta$ sont manifestement δ solutions distinctes.

b) Le lemme chinois donne un isomorphisme d'anneaux $\mathbb{Z}/N\mathbb{Z} \simeq \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z}$. Donc l'équation est équivalente à $a_1^t \equiv 1 \pmod{p}$ et $a_2^t \equiv 1 \pmod{q}$. À chaque couple de solutions (a_1, a_2), l'isomorphisme du lemme chinois associe un unique élément de $\mathbb{Z}/N\mathbb{Z}$ solution de l'équation originelle. Les solutions mod N sont donc en bijection avec le produit cartésien des solutions mod p et mod q ; il y en a donc $\mathrm{pgcd}(p-1, t)\,\mathrm{pgcd}(q-1, t)$ d'après a).

2) a) p étant impair, $p - 1$ est pair ; comme t est impair, on a

$$\mathrm{pgcd}(p - 1, t) = \mathrm{pgcd}((p - 1)/2, t) \leqslant (p - 1)/2.$$

De même $\mathrm{pgcd}(q - 1, t) \leqslant (q - 1)/2$. Donc

$$\mathrm{pgcd}(p - 1, t)\,\mathrm{pgcd}(q - 1, t) \leqslant \frac{(p - 1)}{2}\frac{(q - 1)}{2}.$$

b) S'il n'y a pas de solution, il n'y a rien à démontrer. Sinon, soit α une solution particulière, $\alpha^t = b$. Comme $b \in (\mathbb{Z}/N\mathbb{Z})^*$, on a $\alpha \in (\mathbb{Z}/N\mathbb{Z})^*$ et on en déduit $(a/\alpha)^t = 1$, soit de nouveau $\mathrm{pgcd}(p - 1, t)\,\mathrm{pgcd}(q - 1, t)$ possibilités pour a/α, donc pour a.

3) a) Comme $cd \equiv 1 \pmod{\varphi(N)}$, on a $cd - 1 = k\varphi(N)$, pour un entier k. Donc $a^{t2^e} = a^{cd-1} = a^{\varphi(N)k} = 1$, d'après le théorème d'Euler (on a supposé $a \in (\mathbb{Z}/N\mathbb{Z})^*$).

Soit o_p (resp. o_q) l'ordre de a modulo p (resp. q). L'existence d'un i comme demandé est équivalente à $v_2(o_p) \neq v_2(o_q)$. En effet, soit $v = v_2(o_q)$, on peut supposer $v \geqslant v_2(o_p)$ en échangeant p et q, soit $a^{t2^v} = 1$:

- si on a égalité et $v_2(o_p) = v$, alors $a^{t2^{v-1}} = -1$ et i n'existe pas.
- sinon on a $a^{t2^{v-1}} \equiv 1 \pmod{p}$ mais $\not\equiv 1 \pmod{q}$, donc $a^{t2^{v-1}} \neq \pm 1 \pmod{N}$ et on peut poser $i = v$

b) Si x est comme dans l'énoncé, on a $x^2 = 1$, $x \neq \pm 1$ dans $\mathbb{Z}/N\mathbb{Z}$. Comme dans la méthode de factorisation de Dixon, on en déduit que $\operatorname{pgcd}(N, x-1)$ est un facteur $\neq 1, N$ de N (c'est un diviseur par définition ; si $= N$, on a $N \mid x-1$ soit $x \equiv 1$, absurde ; si $= 1$, $N \mid (x-1)(x+1) \Rightarrow N \mid x+1$ par lemme de Gauss, et $x \equiv -1$, absurde). Il vaut donc p ou q ; le quotient $N/\operatorname{pgcd}(N, x-1)$ donne l'autre facteur.

4)

Algorithme 1
Entrée : c, d, N. Sortie : un diviseur premier de N.

(1) Calculer $e \geqslant 0$, t impair, tels que $cd - 1 = 2^e t$.
(2) Tirer a au hasard dans $[1, N-1]$
(3) Si $\operatorname{pgcd}(a, N) \neq 1$, retourner $\operatorname{pgcd}(a, N)$ (= p ou q).
(4) Calculer $x = a^t$ dans $\mathbb{Z}/N\mathbb{Z}$. Si $x = \pm 1$, revenir en (2).
(5) Tant que $x^2 \neq 1$, répéter $x \leftarrow x^2$.
(6) Retourner $\operatorname{pgcd}(N, x-1)$.

Si on suppose $0 \leqslant c, d \leqslant N$, le calcul de e et t se fait en temps $\widetilde{O}(\log N)$. Pour chaque a tiré au hasard, le coût de la suite de l'algorithme est $\widetilde{O}(\log N)^2$ (on utilise le fait que $2^e t \leqslant cd \leqslant N^2$, donc $\log t = O(\log N)$ et $e = O(\log N)$). On a plus d'une chance sur 2 de succès à chaque essai. Le coût moyen de cet algorithme est donc $\widetilde{O}(\log N)^2$.

5) Factoriser N permet de casser un système RSA reposant sur N : on calcule $\varphi(N) = (p-1)(q-1)$, puis $d = c^{-1}$ modulo $\varphi(N)$ par l'algorithme d'Euclide étendu. On vient de voir que la connaissance de $d \leqslant N$ permet de factoriser N en temps essentiellement quadratique, ce qui est presque la réciproque voulue. Ceci dit, rien n'assure que casser un système RSA nécessite de calculer d (c'est l'opérateur $x \mapsto x^d$ qu'il s'agit de découvrir, pas l'entier d). On ne peut donc pas en déduire l'équivalence demandée.

Exercice II –

1) a) L'ordre de x est un diviseur de l'ordre de G : les valeurs possibles sont incluses dans les $\prod p_i^{f_i}$, où $0 \leqslant f_i \leqslant e_i$ (on a égalité si et seulement si G est cyclique $\Rightarrow$ il existe x d'ordre d pour tout $d \mid n$).

b) Comme $x_1^{p_1^{e_1}} = x^n = 1$, l'ordre de x_1 divise $p_1^{e_1}$, dont les diviseurs sont les p_1^k, $0 \leqslant k \leqslant e_1$. Pour le déterminer :

Algorithme 2
Entrée : x_1, p_1, e_1. Sortie : l'ordre de x_1.

(1) Poser $y \leftarrow x_1$.
(2) Pour $i = 0, \ldots, e_1 - 1$, effectuer
  (a) si $y = 1$, returner p^i. [*Ici*, $y = x_1^{p^i}$.]
  (b) calculer $y \leftarrow y^{p_1}$ par exponentiation binaire.
(3) Returner p_1^e

c) Soit o l'ordre de x et o_1 l'ordre de $x_1 = x^{q_1}$. Puisque $x_1^o = x^{oq_1} = 1$, on a $o_1 \mid o$; puisque $x^{q_1 o_1} = x_1^{o_1} = 1$, on a $o \mid q_1 o_1$. Comme q_1 n'est pas divisible par p_1, on en déduit que $v_{p_1}(o) = v_{p_1}(o_1)$.

2) Pour chaque i, calculer $q_i = n/p_i^{e_i}$, $x_i = x^{q_i}$, et o_i l'ordre de x_i grace à l'algorithme ci-dessus. L'ordre de x est le produit des o_i.

Le calcul de chaque x_i coûte $O(\log q_i) = O(\log n)$ multiplication dans G. Le calcul de o_i coûte $O(e_i \log p_i) = O(\log n)$ dans le cas le pire (correspondant à $o_i = p_i^{e_i}$). Le nombre de p_i, diviseurs premiers distincts de n, est $O(\log n)$. La complexité algébrique est donc $O(\log n)^2$ multiplications dans G.

Exercice III –

1) a) On trouve $r_i = 10^i a$ dans $\mathbb{Z}/b\mathbb{Z}$. Noter que r_i est un entier compris entre 0 et $|b| - 1$, donc le connaître modulo b le détermine complètement.

b) On a $r_i = r_j$ ssi ils sont égaux mod b, ssi $a(10^i - 10^j) \equiv 0 \pmod b$. Comme $\operatorname{pgcd}(a, b) = \operatorname{pgcd}(10, b) = 1$, on sait que 10 et a sont inversibles mod b, soit $10^{i-j} \equiv 0 \pmod b$. On a égalité ssi l'ordre de 10 dans $(\mathbb{Z}/b\mathbb{Z})^*$ divise $i - j$.

2) On a l'égalité requise ssi l'ordre de 10 divise k (indépendamment de $i_0 \geqslant 0$).

3) q_0 est la partie entière de a/b, les autres q_i sont les chiffres du développement décimal de la partie fractionnaire de a/b. Par construction des r_i (r_{i+1} ne dépend que de r_i et de la constante b), on voit que si $r_i = r_{i+k}$, alors $q_{i+\ell} = q_{i+k+\ell}$ pour tout $\ell \geqslant 0$. On vient de voir que ceci se produit ssi l'ordre de 10 divise k.

4)

```
periode := proc(a, b)
  local r0, r, p;
  r0 := irem(a, b); r := r0;
  for p do
    r := irem(10*r, b);
    if (r = r0) then RETURN(p) fi;
  od:
end:
```

5) La période est majorée par $\varphi(b) \leqslant b \leqslant N$, qui majore le nombre d'itérations. Chaque itération se fait en temps $\widetilde{O}(\log N)$ (les opérandes sont tous $\leqslant 10N$) ; soit $\widetilde{O}(N)$ au total.

La méthode de l'exercice II est nettement plus efficace :

- si on suppose la factorisation de b connue, il suffit de $\widetilde{O}(\log N)^2$ multiplications dans $\mathbb{Z}/b\mathbb{Z}$, de complexité binaire $\widetilde{O}(\log N)$, soit $\widetilde{O}(\log N)^3$ au total.
- sinon, la complexité est dominée par le coût de la factorisation de b ; par exemple, par divisions successives par les entiers $\leqslant b^{1/2}$, d'où une complexité binaire $\widetilde{O}(N^{1/2})$.