Université de Rennes 1 — Licence 1ère année
UFR Mathématiques — module A04
Feuille de TD n°6 — Année 2004-2005

Utilisation des applications dans la vie courante

I - Codage d'un caractère alpha-numérique

On rappelle que $\{0,1\}^4 = \{0,1\} \times \{0,1\} \times \{0,1\} \times \{0,1\}$ ou, en d'autres termes que :

$$\{0,1\}^4 = \big\{(x_1,x_2,x_3,x_4)\,|x_0 \in \{0,1\}, x_1 \in \{0,1\}, x_2 \in \{0,1\}, x_3 \in \{0,1\}\big\}.$$

On considère l'application f définie par

$$f : \left[\begin{array}{c} \{0,1\}^4 \longrightarrow \{0,1,2,\ldots,14,15\} \\ (x_0,x_1,x_2,x_3) \longmapsto 2^3\,x_3 + 2^2\,x_2 + 2^1\,x_1 + 2^0\,x_0 = 8\,x_3 + 4\,x_2 + 2x_1 + x_0 \end{array}\right.$$

1) Montrer que f est bijective.

2) Comment s'obtient l'antécédent d'un élément x de $\{0,1,\ldots,15\}$?

3) Soit $n \in \mathbb{N}^*$. Généraliser à une application de $\{0,1\}^n$ dans $\{0,1,\ldots,2^n-1\}$.

Ainsi, tout entier naturel plus petit que $2^n - 1$ peut se coder par une suite de n 0 ou 1. Les informaticiens parlent de "bits". Un champ de 8 bits constitue un "octet" (ou "byte" en anglais). L'octet permet de coder $2^8 = 256$ termes.

La mémoire de l'ordinateur conserve toutes les données sous forme numérique. Il n'existe pas de méthode pour stocker directement les caractères. Il est donc intéressant que chaque caractère possède son équivalent en code numérique. On doit donc associer de manière unique chaque lettre de l'alphabet à un entier, puis coder cet entier en bits. La bijection la plus naturelle est d'associer 1 à A, 2 à B, jusqu'à 26 à Z, mais il existe un codage plus adapté au besoin de l'informatique : le codage **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange, apparu dans les années 60).

Le code ASCII de base représentait les caractères sur 7 bits (c'est-à-dire 128 caractères possibles, de 0 à 127).

- Les codes 0 à 31 ne sont pas des caractères. On les appelle caractères de contrôle car ils permettent de faire des actions telles que le retour à la ligne ou un bip sonore.
- Les codes 65 à 90 représentent les majuscules.
- Les codes 97 à 122 représentent les minuscules. (Il suffit de modifier le 6ème bit pour passer de majuscules à minuscules, c'est-à-dire ajouter 32 au code ASCII en base décimale)

Le code ASCII a été mis au point pour la langue anglaise, il ne contient donc pas de caractères accentués, ni de caractères spécifiques à une langue. Pour coder ce type de caractère il faut recourir à un autre code. Le code ASCII a donc été étendu à 8 bits (un octet) pour pouvoir coder plus de caractères (on parle d'ailleurs de code ASCII étendu).

Le code **Unicode** est un système de codage des caractères sur 16 bits mis au point en 1991. Il permet de représenter n'importe quel caractère par un code sur 16 bits, indépendamment de tout système de programmation ou langage de programmation. Il regroupe ainsi la quasi-totalité des alphabets (chinois, cyrillique, hébreu, grec,...) et est compatible avec le code ASCII.

II - Code correcteur d'erreur

II-1 Le bit de parité

On considère l'application f définie par

$$f : \left[\begin{array}{c} \{0,1\}^4 \longrightarrow \{0,1\}^5 \\ (x_0,x_1,x_2,x_3) \longmapsto (x_0,x_1,x_2,x_3,x_4) \end{array}\right.$$

où $x_4 = 0$ si $x_0 + x_1 + x_2 + x_3$ est pair et $x_4 = 1$ si $x_0 + x_1 + x_2 + x_3$ est impair.
1) L'application f est-elle surjective ? Injective ?

2) Montrer que l'image de f est le sous-ensemble F de $\{0,1\}^5$ défini par

$$F = \big\{(x_0, x_1, x_2, x_3, x_4) \in \{0,1\}^5 \,|\, x_0 + x_1 + x_2 + x_3 + x_4 \text{ est pair}\big\}.$$

Cette application, ou du moins ses généralisations, appartient à la théorie des **codes correcteurs d'erreurs**. Le contexte est le suivant : Alice veut envoyer un message à Bob. On suppose, dans notre cas, que le message en question est un élément m de $\{0,1\}^4$. Le problème qui se pose concerne la fiabilité du canal de transmission : il se peut qu'un 0 (respectivement un 1) soit transformé en un 1 (respectivement en un 0). Par exemple, le message $m = (0,1,1,1)$ peut être altéré en $m' = (0,1,0,1)$ si le bit x_2 est modifié. Comment Bob peut-il savoir si le message m' qu'il reçoit est celui que lui a envoyé Alice ? Pour répondre à ce genre de problème, on utilise l'application f. Au lieu d'envoyer m à Bob, Alice envoie $f(m)$. Dans l'exemple précédent, Alice n'envoie pas $m = (0,1,1,1)$ mais $f(m) = (0,1,1,1,1)$.

3) Montrer que, si on suppose qu'aucune erreur de transmission ne survient, alors Bob peut déduire m du message reçu.

4) Quelle propriété de f utilise-t-on ?

On suppose maintenant qu'un unique bit du message m a été altéré lors de la transmission.
5) Comment Bob s'en aperçoit-il ?

L'ajout du bit de parité permet donc de détecter lorsqu'une seule erreur se produit. Par contre, on ne peut pas localiser l'erreur, et donc la corriger. Par exemple, si Alice envoie $f(m) = (0,1,1,1,1)$ et s'il se produit une erreur au niveau du troisième bit, Bob reçoit le message $r = (0,1,0,1,1)$. A priori, s'il fait la liste des erreurs possibles sur chaque bit, il peut seulement dire que :

$$f(m) \in \big\{(1,1,0,1,1), (0,0,0,1,1), (0,1,1,1,1), (0,1,0,0,1), (0,1,0,1,0)\big\}.$$

On note $\overline{x}$ le bit $1 - x$.
6) Montrer que

$$m \in f^{-1}\Big(\big\{(\overline{x}_0, x_1, x_2, x_3, x_4), (x_0, \overline{x}_1, x_2, x_3, x_4), (x_0, x_1, \overline{x}_2, x_3, x_4), (x_0, x_1, x_2, \overline{x}_3, x_4), (x_0, x_1, x_2, x_3, \overline{x}_4)\big\}\Big)$$

II-2 Le code à répétition

Considérons l'application suivante

$$f : \left[\begin{array}{l} \{0,1\}^2 \longrightarrow \{0,1\}^6 \\ (x_0, x_1) \longmapsto (x_0, x_1, x_0, x_1, x_0, x_1) \end{array}\right.$$

7) L'application f est-elle surjective ? Injective ?

Bob reçoit d'Alice le message $r = (1,0,1,1,1,0)$.
8) Montrer que r n'appartient pas à l'image de f. Interprétation ?

On suppose qu'il ne s'est produit qu'une seule erreur au cours de la transmission. On note

$$\overline{r} = \big\{(c_0, c_1, \ldots, c_5) \in \{0,1\}^6 \,|\, \exists i_0 \in \{0, \ldots, 5\},\ c_{i_0} = \overline{r}_{i_0} \text{ et } \forall i \neq i_0,\ c_i = r_i\big\}.$$

9) Calculer $f^{-1}(\overline{r})$. Interpréter le résultat.

Cet argument, adapté au cas général, montre que le code permet de repérer lorsqu'il se produit **une** erreur, et de la corriger.
10) Est-il possible de corriger deux erreurs ?

II-3 Le code ISBN

Un numéro ISBN (au dos de chaque livre) est composé de dix chiffres qu'on peut décomposer en quatre blocs. Considérons l'exemple suivant : 3 540 60226 7. Le premier bloc 3 indique le pays, le deuxième 540 l'éditeur, le troisième 60226 la référence du livre chez l'éditeur ; quant au quatrième bloc, il s'agit d'un chiffre choisi de manière à ce que tous les chiffres vérifient une certaine égalité algébrique.
Plus précisément, si on note $x_1x_2\dots x_{10}$ un code ISBN, x_{10} est l'entier tel que $x_1+2x_2+3x_3+\dots+10x_{10}$ soit divisible par 11.
Le code ISBN utilise donc l'application injective et non surjective suivante

$$f:\left[\begin{array}{c}\{0,1,\dots,9\}^9\longrightarrow\{0,1,\dots,9\}^{10}\\(x_1,x_2,\dots,x_9)\longmapsto(x_1,x_2,\dots,x_{10})\end{array}\right.$$

où x_{10} est défini par la relation ci-dessus.
11) Montrer que le code ISBN permet de détecter une erreur, mais pas de la corriger.

12) Montrer que le code ISBN permet de corriger un effacement (un blanc est apparu à la place d'un chiffre).

III - La cryptographie

Alice cherche toujours à transmettre un message à Bob, mais, cette fois-ci, on suppose qu'aucune erreur de transmission ne survient. L'objectif est ici différent : on souhaite que seul Bob puisse lire le message.
On note $\mathscr{M}$ l'ensemble des messages qu'Alice est susceptible d'envoyer. Ici, on prendra $\mathscr{M}=\mathscr{A}^{10}$ où $\mathscr{A}=\{A,B,\dots,Z\}$. Un message est donc uniquement un mot de 10 lettres.
L'opération de chiffrement dépendra d'une clé K qui fournira une application f_K de $\mathscr{M}$ dans lui-même. On exige que f_K soit bijective. L'application f_K^{-1} est **l'application de déchiffrement**.

III-1 Le chiffrement de César

La clé K est simplement un élément de $\mathscr{A}$. L'application consiste juste à faire opérer, sur chaque lettre du message, un décalage de lettres égal au rang de K dans l'alphabet. Ainsi, si $K=$"C", on fait un décalage de trois lettres. L'image du message $M=$"ALLEZ,ALLEZ !" est donc $f_K(M)=$"DOOHC,DOOHC !".
Si Bob connait la clé utilisée par Alice, il peut calculer f^{-1}. Dans l'exemple précédent, pour $K=C$, l'application de déchiffrement s'obtient en opérant un décalage de 3 lettres vers la gauche (ou ce qui revient au même, quoique moins rapide, à faire un décalage de 23 lettres vers la droite).
1) Si la clé vaut successivement A, M, puis Z, donner le message chiffré de "C'EST SECRET".

On comprend aisément que ce type de chiffrement n'est pas très sûr ! Il suffit pour essayer de déchiffrer un message d'essayer les fonctions de déchiffrements associées aux 26 clés possibles.

En cryptographie, il existe une attaque classique élémentaire pour savoir si une application de chiffrement n'est pas efficace, à savoir **l'analyse des fréquences**. Lorsque l'on dispose d'un texte chiffré suffisamment long, on remplace les lettres qui apparaissent le plus souvent dans le texte chiffré par des "E" (lettre qui apparait le plus fréquemment dans un texte en français), puis c'est au tour du "A" la deuxième lettre la plus fréquente,etc. jusqu'à avoir suffisamment d'informations pour en déduire le texte clair. A noter que Georges Perec a écrit un livre "La Disparition" qui n'emploie pas un seul mot contenant la voyelle "E", livre traduit en anglais par Gilbert Adair en suivant le même principe.
2) Sachant que le message "UWJHJIJSYJ" a été obtenu par le chiffrement de César, déchiffrer le et donner sa clé de déchiffrement.

III-2 Le chiffrement affine

La clé est cette fois constituée de deux entiers $a\in\{0,1,\dots,25\}$ et $b\in\{0,1,\dots,25\}$. On considère d'abord l'application $f:\ \{0,1,\dots,25\}\longrightarrow\{0,1,\dots,25\}$, où $f(x)$ est le reste de la division euclidienne de $ax+b$ par 26.
3) Montrer que, si $a=1$, f est une bijection.

4) f est-elle une bijection si $a=2$?

5) Si $a=5$ et $b=3$, calculer $f^{-1}(\{4\})$.

On admet que f est une bijection si a n'est pas divisible par 13 ou par 2. On suppose dans la suite que a satisfait cette condition.
Venons-en à l'application de chiffrement :

$$F : \left[\begin{array}{l} \{0,1,\ldots,25\}^{10} \longrightarrow \{0,1,\ldots,25\}^{10} \\ (x_1,x_2,\ldots,x_{10}) \longmapsto \big(f(x_1),f(x_2),\ldots,f(x_{10})\big) \end{array} \right.$$

6) Montrer que F est une bijection.

On identifie l'alphabet à $\{0,1,\ldots,25\}$ (i.e ; $A = 0$, $B = 1$ etc.). Le message "C'EST SECRET" se code 2,4,18,19,18,4,2,17,4,19.
7) Que vaut $F(M)$ si $a = 7$ et $b = 16$? Quel est le message alphabétique correspondant ?

III-3 Le chiffrement de Vernam

Pour éviter des attaques du type "analyse de fréquence", on peut augmenter la taille de la clé : au lieu de n'avoir que a et b, on définit plusieurs entiers, le cas extrême se produisant lorsque la taille de la clé est égale à la taille du message. Ici, pour des messages de dix lettres, on prend $K = (k_1,k_2,\ldots,k_{10}) \in \{0,1,\ldots,25\}^{10}$ et on définit l'application de déchiffrement suivante :

$$F : \left[\begin{array}{l} \{0,1,\ldots,25\}^{10} \longrightarrow \{0,1,\ldots,25\}^{10} \\ (x_1,x_2,\ldots,x_{10}) \longmapsto (y_1,y_2,\ldots,y_{10}) \end{array} \right.$$

où, pour tout $i \in \{1,\ldots,10\}$, y_i est le reste de la division de $x_i + k_i$ par 26. C'est le chiffrement de Vernam.
8) Montrer que F est une bijection.

On prend la clé $K = (10,1,23,28,2,3,17,0,12,11)$ et on identifie l'alphabet à $\{0,1,\ldots,25\}$.
9) Chiffrer le message "C'EST SECRET". Que vaut $F(M)$? Quel est le message alphabétique correspondant ?

III-4 La cryptographie dite à clé publique

On suppose que Alice dispose d'une clé secrète D_A et d'une clé publique C_A. C_A est donc connue de tout le monde, alors que seule Alice connaît D_A.
Associée à C_A, Alice dispose donc d'une application de chiffrement F_{C_A} qui est publique. Elle dispose également d'une application de chiffrement F_{D_A} qui est secrète. Ces applications de déchiffrement sont construites de telle manière que f_{D_A} soit la bijection réciproque de la bijection f_{C_A} mais que cette bijection réciproque soit impossible à déterminer sans un peu d'information sur la manière dont Alice a choisi D_A.
Le principe de construction de ces clefs est de choisir deux nombres premiers p et q qui constituent la clé secrète, la clé publique étant le produit N de ces deux nombres premiers. Or, factoriser un entier N en produits d'entiers premiers demande essentiellement de faire la division de N par tous les entiers premiers inférieurs à $\sqrt{N}$ jusqu'à trouver un reste nul. Plus N est grand, plus ces calculs sont longs. Pour des transactions bancaires importantes, N est de l'ordre de 10^{308}. Il a été estimé que les efforts combinés de cent millions d'ordinateurs prendraient plus de mille ans pour trouver p et q pour un tel N.
De même, Bob dispose d'une clé secrète D_B et d'une clé publique C_B.
Pour des besoins de confidentialité, Alice veut transmettre un message M que seul Bob pourra déchiffrer. Pour cela, elle transmet $f_{C_B}(M)$. Bob est le seul à connaître l'application de déchiffrement f_{D_B}. Il calcule donc $f_{D_B}\big(f_{C_B}(M)\big) = M$.
Pour des besoins d'authenticité, Alice veut transmettre un message M de sorte que Bob soit certain qu'Alice en est l'expéditrice. Pour cela, elle transmet $f_{D_A}(M)$. Comme tout le monde, Bob connaît l'application de chiffrement f_{C_A}. Il peut donc calculer $f_{C_A} \circ f_{D_A}(M) = M$. Il peut certes prendre connaissance du message d'Alice, mais, de plus, comme Alice est la seule à connaître la clé D_A, il est sûr que le message provient bien d'Alice.
10) Comment envoyer M en assurant confidentialité et authenticité en même temps ?

Pour en savoir plus : Histoire des codes secrets de Simon Singh