Option Informatique en Spé MP et MP*
Le problème du monnayeur : le corrigé

Il s'agit du corrigé du sujet que j'avais initialement conçu à l'automne 2000, pour le Concours Centrale-Supélec ; après une longue décantation et quelques remaniements sauvages indépendants de ma volonté, ce sujet a été soumis à la sagacité des candidats en mai 2002.

La numérotation des questions est continue de 1 à 34. Par rapport au sujet du concours précité, on trouvera quelques questions en plus, et surtout la partie (qui n'avait pas été retenue) consacrée à l'algorithme de Pearson.

J'ai ajouté les numéros des questions qui apparaissent dans la version donnée aux candidats ; la question IV.H de cette version n'est pas corrigée ici. La réponse à la question III.F (canonicité du système «euro») est donnée en annexe ; je remercie Laurent Chéno de m'avoir transmis cette preuve, fort technique, et que je n'aurais certainement pas eu le courage d'établir ...

La réponse à la question IV.D est donnée dans une remarque qui suit la question 19. Par ailleurs, les énoncés des questions III.E et IV.F ne sont pas exactement ceux des questions 12 et 21 respectivement ; les réponses proposées ici ne peuvent donc s'appliquer telles quelles.

Il manque les réponses aux question 16 et 34.

Le problème du monnayeur

Question 1 **(I.A)** • Un système est une liste strictement décroissante, dont le dernier élément est 1. Donc :

- une liste de longueur au moins 2 est un système ssi son premier élément majore strictement le deuxième, et si la queue de liste est à son tour un système ;
- une liste de longueur 1 est un système ssi son unique élément est 1 ;
- la liste vide n'est pas un système.

La traduction de cette spécification en Caml est immédiate :

```
let rec est_un_systeme = function
 | a::b::q -> a>b & est_un_systeme (b::q)
 | [1] -> true
 | [_] | [] -> false
```

Représentations de poids minimal

Question 2 **(II.A)** • En utilisant x pièces de valeur $c_m = 1$, nous obtenons une représentation de x (de poids maximal) donc $M(x) \leqslant x$. Soit k une représentation minimale de x : $x = k \cdot c = \sum_{1 \leqslant i \leqslant m} k_i c_i \leqslant \sum_{1 \leqslant i \leqslant m} k_i c_1 = \|k\| c_1$; donc $M(x) = \|k\| \geqslant x/c_1$; mais $\|k\| \in \mathbb{N}^*$, donc $M(x) \geqslant \lceil x/c_1 \rceil$.

Question 3 **(II.B)** • $(1,0,1)$ et $(0,2,0)$ sont deux représentations minimales de $x = 4$ dans le système $c = (3,2,1)$.

Question 4 **(II.C)** • Toute représentation minimale $k = (k_i)_{1 \leqslant i \leqslant m}$ de $x - c_j$ fournit une représentation k' de x définie par $k'_i = k_i$ si $i \neq j$ et $k'_j = k_j + 1$. Son poids est $\|k'\| = 1 + M(x - c_j)$; donc $M(x) \leqslant 1 + M(x - c_j)$.

Question 5 **(II.D)** • Supposons $M(x) = 1 + M(x - c_j)$. Soit k une représentation k' minimale de $x - c_j$; en remplaçant k_j par $k_j + 1$, nous obtenons une représentation de x faisant intervenir c_j, et qui est minimale puisque son poids est $1 + M(x - c_j) = M(x)$.

• Réciproquement, soit k une représentation minimale de x faisant intervenir c_j : nous avons donc $k_j \geqslant 1$. En remplaçant k_j par $k_j - 1$, nous obtenons une représentation de $x - c_j$, de poids $M(x) - 1$. Donc $M(x - c_j) \leqslant M(x) - 1$. Mais le résultat de la question 4 donne $M(x - c_j) \geqslant M(x) - 1$; d'où l'égalité demandée.

Question 6 **(II.E)** • Soit $j \in [\![s, m]\!]$ tel que $M(x - c_j) = \min_{s \leqslant i \leqslant m} M(x - c_i)$. La décomposition correspondante de $x - c_j$ nous donne une décomposition de x de poids $1 + M(x - c_j)$; donc $M(x) \leqslant 1 + M(x - c_j)$ et à plus forte raison $M(x) \leqslant 1 + \min_{s \leqslant j \leqslant m} M(x - c_j)$.

• Pour établir l'inégalité inverse, observons une représentation minimale de x. Soit j un indice tel que $k_j \neq 0$ dans cette représentation. Nous avons certainement $j \geqslant s$, par définition de s. En diminuant k_j de 1, nous obtenons une décomposition de $x-c_j$, de poids $M(x)-1$. Donc $M(x-c_j) \leqslant M(x)-1$, puis $1+M(x-c_j) \leqslant M(x)$ et par conséqueent $1 + \min_{s\leqslant j\leqslant m} M(x-c_j) \leqslant M(x)$.

Question 7 **(II.F)** • L'idée est de construire la liste par valeurs croissantes de y. Nous avons déjà $M(1) = 1$. Supposons obtenues les valeurs de $M(i)$ pour $1 \leqslant i < y$. Appliquons la formule de la question précédente : déterminons le plus petit indice j tel que $c_j \leqslant y+1$; ensuite, pour chaque $i \in [\![j, m]\!]$, calculons $x - c_i$, puis $1 + M(x - c_i)$. Il reste à déterminer la plus petite des $m - j + 1$ valeurs ainsi construites. Il est clair que le coût de l'obtention de $M(y + 1)$ est un $\mathcal{O}(m)$; donc le coût total est un $\mathcal{O}(mx)$.

Question 8 **(II.G)** • Appliquons la formule établie à la question 6. La technique mise en œuvre est connue sous le nom de *programmation dynamique* : on mémorise dans un vecteur les valeurs de $M(y)$, au fur et à mesure de leur calcul. Le vecteur, une fois rempli, est converti en liste. Notons que nous sortons du cadre strict de la programmation fonctionnelle, puisque nous avons recours à une structure de données mutable ; en fait, chaque «case» du vecteur n'est modifiée qu'une fois, pour recevoir sa valeur définitive.

Les vecteurs étant indexés à partir de 0, nous construisons un vecteur v de longueur $x + 1$, dont nous oublierons la première composante.

Soit $y \in [\![1, x]\!]$. Supposons que les valeurs de $M(i)$, pour $0 \leqslant i < y$, ont été placées dans les cases correspondantes du vecteur v. Enlevons de la liste c les valeurs supérieures à y ; nous obtenons une liste $(c_i)_{s\leqslant i\leqslant m}$, dont nous déduisons la liste $(y - c_i)_{s\leqslant i\leqslant m}$. Appliquons alors la formule établie à la question 6 ; la valeur obtenue est rangée dans la case y du vecteur.

```
let rec dropwhile p = function
 | t::q when p t -> dropwhile p q
 | l -> l ;;

let v_poids_minimaux x c =
 let v = make_vect (x+1) 0 in
 for y = 1 to x do
  let c' = map (fun t -> v.(y-t)) (dropwhile (prefix < y) c) in
  v.(y) <- 1 + it_list min y c'
 done ; v ;;

let poids_minimaux x c = list_of_vect(v_poids_minimaux x c) ;;
```

La fonction `v_poids_minimaux` sera utilisée à nouveau dans la question 23.

L'algorithme glouton

Question 9 **(III.A)** • Utilisons l'accélérateur évoqué dans l'énoncé. L'écriture du programme est immédiate, en notant que la représentation gloutonne de x dans le système $c = (c_i)_{1\leqslant i\leqslant m}$ est la liste $k = (k_i)_{1\leqslant i\leqslant m}$ définie par les deux conditions suivantes :

- $k_1 = \lfloor x/c_1 \rfloor$;
- $(k_i)_{2\leqslant i\leqslant m}$ est la représentation gloutonne de $x - k_1c_1$ dans le système $(c_i)_{2\leqslant i\leqslant m}$.

```
let rec glouton x = function
 | [] -> []
 | t::q -> (x/t)::(glouton (x mod t) q) ;;
```

Question 10 **(III.C)** • Soit $x > 0$; notons q et r le quotient et le reste dans la division euclidienne de x par c_1. Soit $k = (k_1, k_2)$ une décomposition : si $k_2 < c_1$, alors $k_1 = q$, $k_2 = r$, et k est la décomposition gloutonne. Sinon, il existe une décomposition $k' = (k_1 + 1, k_2 - c1)$ de poids $\|k'\| = k_1 + k_2 + 1 - c_1 < \|k\|$ puisque $c_1 > 1$. Ceci montre que la décomposition gloutonne est minimale.

Question 11 **(III.D)** • Le système $(4, 3, 1)$ n'est pas canonique : la décomposition gloutonne de $x = 6$ est $(k_1 = 1, k_2 = 0, k_3 = 2)$, de poids 3, tandis que la décomposition $(k_1 = 0, k_2 = 2, k_3 = 0)$ a un poids 2.

Question 12 **(III.E modifiée)** • Considérons le système $c = (9, 4, 1)$: il vérifie bien la condition de l'énoncé. Mais il n'est pas canonique, car $x = 12$ admet pour décomposition gloutonne $(1, 0, 3)$, de poids 4 ; or il existe une décomposition de poids 3, à savoir $(0, 3, 0)$.

• Nous pouvons assez facilement établir qu'un système c vérifiant la condition $c_i = 2c_{i+1}$ pour $1 \leqslant i < m$ est canonique ; en fait, nous allons donner la preuve pour c vérifiant $c_i = \lambda c_{i+1}$ pour $1 \leqslant i < m$, où $\lambda \geqslant 2$ est fixé. Soit $x \geqslant 1$; considérons une décomposition k de x. S'il existe un indice $i \geqslant 2$ tel que $k_i \geqslant \lambda$, nous pouvons remplacer k_i par $k_i - \lambda$ et k_{i-1} par $k_{i-1} + 1$ pour obtenir une représentation de poids strictement inférieur, donc k n'est pas minimale. Si $0 \leqslant k_i < \lambda$ pour $2 \leqslant i \leqslant m$, alors le mot $c_2 \ldots c_m$ est l'écriture en base $s = \lambda$ de $x - k_1 c1$, et $k_1 = \lfloor x/c_1 \rfloor = (x - s)/c_1$, ce qui est précisément la décomposition gloutonne de x.

Question 13 **(III.G)** • La décomposition gloutonne de 48 est $(k_1 = 1, k_2 = 0, k_3 = 1, k_4 = 1, k_5 = 0, k_6 = 0)$ de poids 3, alors qu'il existe une décomposition $(k_1 = 0, k_2 = 2, k_3 = 0, k_4 = 0, k_5 = 0, k_6 = 0)$ de poids 2.

Question 14 **(non retenue)** • Observons le monnayeur d'un distributeur de café : il reçoit des pièces, qu'il va trier et répartir dans autant de bacs qu'il y a de valeurs faciales. Si l'un de ces bacs est presque plein, le monnayeur peut avoir avantage à utiliser de préférence les pièces de ce bac pour rendre la monnaie.

Question 15 • Soit $i \in [\![2, m]\!]$; nous avons $k_i c_i \leqslant x - \sum_{1 \leqslant j < i} k_j c_j < c_{i-1} = c_i + c_{i+1} < 2c_i$, donc $k_i < 2$.

• Soit $i \in [\![2, m]\!]$ tel que $k_i = 1$. Alors $c_i + k_{i+1} c_{i+1} \leqslant x - \sum_{1 \leqslant j < i} k_j c_j < c_{i-1} = c_i + c_{i+1}$ donc $k_{i+1} = 0$.

Question 16 • **À FAIRE !** (le système de FIBONACCI est canonique)

L'algorithme de Kozen et Zaks

Question 17 **(IV.A)** • S'il existe un contre-exemple, il en existe un minimal. Notons-le x. Il est clair que $x > c_{m-2} + 1$: le système (c_{m-1}, c_m) étant canonique, la décomposition gloutonne de tout élément de $[\![1, c_{m-2} - 1]\!]$ est minimale ; celle de c_{m-2} a un poids 1 ; et celle de $c_{m-2} + 1$, de poids 2, est manifestement minimale.

• Établissons maintenant la majoration. La décomposition gloutonne de $c_1 + c_2$ est minimale ; supposons acquise l'égalité $G(y) = M(y)$ pour $c_1 + c_2 \leqslant y < x$, et prouvons que nous avons encore $G(x) = M(x)$. Si c_1 intervient dans une décomposition minimale de x, alors $G(x) = 1 + G(x - c_1)$ par définition de G ; $G(x - c_1) = M(x - c_1)$ de par l'hypothèse de récurrence ; et $1 + M(x - c_1) = M(x)$ d'après le résultat de la question 5.

• Si c_1 n'intervient pas dans une décomposition minimale de x, notons i un indice tel que c_i intervienne dans une telle décomposition. Remarquons que $x \geqslant c_1 + c_i$. Le raisonnement commence comme précédemment : $G(x) = 1 + G(x - c_1) = 1 + M(x - c_1)$. Le résultat de la question 5 permet d'affirmer que $M(x - c_1) \leqslant 1 + M(x - c_1 - c_i)$; $M(x - c_1 - c_i) \leqslant G(x - c_1 - c_i)$ par définition de M ; $G(x - c_1 - c_i) = G(x - c_i) - 1$ par définition de G ; $G(x - c_i) = M(x - c_i)$ par hypothèse de récurrence ; $1 + M(x - c_i) = M(x)$, à nouveau grâce au résultat de la question 5 ; et enfin $M(x) \leqslant G(x)$ par définition de M. Par transitivité, nous obtenons $G(x) \leqslant M(x) \leqslant G(x)$, donc $M(x) = G(x)$, ce qui achève la preuve.

Question 18 **(IV.B)** • Pour le système considéré, $m = 3$, $c_{m-2} + 1 = c_1 + 1 = q + 2$ et $c_1 + c_2 = 2q + 1$. Le résultat de la question 17 nous permet de ne tester que les éléments de l'intervalle $[\![q + 3, 2q]\!]$. Observons que $x = 2q$ a pour décomposition gloutonne $(k_1 = 1, k_2 = 0, k_3 = q - 1)$ de poids $q \geqslant 3$; mais x admet aussi la décomposition $(k_1 = 0, k_2 = 2, k_3 = 0)$, de poids 2 : donc $2q$ est un contre-exemple, et le système considéré n'est pas canonique.

• Soit maintenant $x \in [\![q + 3, 2q - 1]\!]$: sa décomposition gloutonne $(k_1 = 1, k_2 = 0, k_3 = x - q - 1)$ est clairement minimale. Donc $2q$ est le plus petit contre-exemple.

Question 19 **(IV.C)** • Choisissons $\alpha(q) = 2q - 2$. Alors $\alpha(q) = q + (q - 2) > q$ car $q \geqslant 3$. La représentation gloutonne de $2q$ est $(k_1 = 1, k_2 = 0, k_2 = 2)$, de poids 3 ; et $2q$ admet une représentation $(k_1 = 0, k_2 = 2, k_3 = 0)$ de poids 2. D'après la question 17, ce contre-exemple est minimal puisque $2q = c_1 + 2$.

► Nous venons de montrer que l'encadrement établi à la question 17 est optimal. Plus précisément : la question 19 montre qu'il existe une infinité de systèmes non canoniques pour lesquels $c_{m-2} + 2$ est le plus petit contre-exemple ; et la question 18 montre qu'il existe une infinité de systèmes non canoniques pour lesquels $c_1 + c_2 - 1$ est le plus petit contre-exemple.

Question 20 **(IV.E)** • Soit x un témoin : il existe donc i tel que $G(x - c_i) < G(x) - 1$. Mais $M(x) \leqslant M(x - c_i) + 1$ (résultat de la question 5) ; d'autre part, $M(x - c_i) \leqslant G(x - c_i)$ par définition de M. Donc $M(x) < G(x)$ par transitivité : x est un contre-exemple.

Question 21 **(IV.F modifiée)** • La valeur 12 est un contre-exemple pour le système $(5, 4, 1)$ puisqu'elle admet la représentation gloutonne $(k_1 = 2, k_2 = 0, k_3 = 2)$ de poids 4, et une représentation $(k_1 = 0, k_2 = 3, k_3 = 0)$ de poids 3. En revanche, cette valeur n'est pas un témoin : vous vérifierez aisément que $G(12 - c_1) = G(7) = 3$, $G(12 - c_2) = G(8) = 4$ et $G(12 - c_3) = G(11) = 4$.

Question 22 **(IV.G)** • Nous allons montrer que, si un contre-exemple n'est pas un témoin, alors il existe au moins un contre-exemple plus petit. Ceci prouvera que le plus petit contre-exemple est nécessairement un témoin. Soient donc x un contre-exemple, et i tel que c_i intervienne dans une représentation minimale de x. Nous aurons $M(x-c_i) = M(x)-1$ (résultat de la question 5), $M(x) < G(x)$ puisque x est un contre-exemple, et $G(x) \leqslant 1 + G(x - c_i)$ puisque x n'est pas un témoin. Donc $M(x - c_i) < G(x - c_i)$: $x - c_i$ est aussi un contre-exemple.

Question 23 **(IV.I)** • On construit la liste des témoins, en restreignant la recherche à l'intervalle $[\![c_{m-2} + 2, c_1 + c_2 - 1,]\!]$; il reste à vérifier si cette liste est vide. La fonction `filtre` est spécifiée comme suit : `filtre p l` construit, à partir de la liste ℓ, une nouvelle liste ne contenant que les éléments qui satisfont le prédicat p.

```
let poids_glouton x l = it_list (prefix +) 0 (glouton x l) ;;

let liste_contre_exemples l =
 let lgr = list_length l in if lgr < 3 then [] else
 let mini = hd(tl(tl(rev l))) and maxi = hd l + hd(tl l) in
 let iv = intervalle mini maxi and pm = v_poids_minimaux maxi l in
 filtre (fun x -> (poids_glouton x l) <> pm.(x)) iv ;;

let kozen_zaks l = liste_contre_exemples l = [] ;;
```

Question 24 **(IV.K)** • Le calcul des valeurs de $G(x)$ a un coût $\mathcal{O}(mc_1)$; les vérifications ont un coût $\mathcal{O}(mc_1)$; or c_1 peut fort bien être exponentiel en m.

Question 25 • Si $0 < r < t - q$, alors $x = s + t - 1$ est un contre-exemple : sa représentation gloutonne est $(k_1 = 1, k_2 = 0, k_3 = t - 1)$ de poids t ; et la représentation $(k_1 = 0, k_2 = q + 1, k_3 = r - 1)$ a pour poids $q + r < t$.

• Réciproquement, supposons le système $(s, t, 1)$ non canonique, et notons x le plus petit contre-exemple. Sa représentation gloutonne est $(1, 0, e)$ avec $1 < e < t$ puisque $s+1 < x < s+t$; ici encore, nous utilisons la question 17. Dans une représentation minimale $k = (k_1, k_2, k_3)$ de x, nous avons nécessairement $k_1 = 0$, mais aussi $k_3 = 0$ sinon $x-1$ serait un contre-exemple plus petit que x. Alors $s = k_2 t - e = (k_2 - 1)t + (t - e)$, avec $0 < s + t - x = (s + t) - (s + e) = t - e < t$. Donc, dans la division euclidienne de s par t, le quotient est $q = k_2 - 1$ et le reste est $r = t - e$. Mais $t - e > 0$; et, comme x est un contre-exemple, $k_2 = \|k\| = M(x) < G(x) = 1 + e$, d'où $r = t - e < t + 1 - k_2 = t - q$. Joli, non?

L'algorithme de Pearson

Question 26 • Raisonnons par récurrence sur x. Pour $x = 1$, le résultat est clair : la représentation gloutonne est le vecteur $(0, \ldots , 0, 1)$; comme c'est la seule, elle est lexicographiquement maximale. Soit $x > 1$; supposons le résultat acquis pour tout $y < x$. Soient g la représentation gloutonne de x, et r une autre représentation. Soit i le plus petit indice tel que $g_i \neq r_i$. Notons $x' = x - \sum_{1 \leqslant j < i} g_j c_j$: $(g_j)_{i \leqslant j \leqslant m}$ est *la* représentation gloutonne de x' dans le système $(c_j)_{i \leqslant j \leqslant m}$, donc $g_i = \lfloor x'/c_i \rfloor$. $(r_j)_{i \leqslant j \leqslant m}$ est *une* représentation de x' dans le système $(c_j)_{i \leqslant j \leqslant m}$: donc $x' \geqslant r_i c_i$. Nous en déduisons $r_i \leqslant g_i$. Comme $r_i \neq g_i$, ceci implique $r_i < g_i$ et donc $r \prec g$.

Question 27 • Notons $\Gamma_c(x) = (g_1, \ldots , g_m)$. Alors $v = (g_1, \ldots , g_{m-1}, g_m + y - x)$ est *une* représentation de y dans le système c. Il est clair que $\Gamma_c(x) \prec v$; et le résultat de la question 26 affirme que $v \preceq \Gamma_c(y)$. Donc $\Gamma_c(x) \prec \Gamma_c(y)$.

Question 28 • Soit u' une représentation de $u \cdot c$: nous avons $u' \cdot c = u \cdot c$, donc $(v - u + u') \cdot c = v \cdot c$. Toutes les composantes de $v - u$ sont positives ou nulles, donc $v - u + u'$ est une représentation de $v \cdot c$; comme $v = \Gamma_c(v \cdot c)$, le résultat de la question 26 nous donne $v - u + u' \preceq v$. Nous ne changeons pas l'ordre lexicographique respectif de deux m-uplets en leur ajoutant un même m-uplet : donc $u' \preceq u$. Ceci montre que, parmi les représentations de $u \cdot c$, u est lexicographiquement maximale, soit $u = \Gamma_c(u \cdot c)$.

Question 29 • La preuve est exactement la même, en ne considérant que les représentations minimales de $u \cdot c$.

Question 30 • Si l'on avait $g_i \omega_i \neq 0$, alors $\xi - c_i$ serait un contre-exemple strictement inférieur à ξ.

Question 31 • Si $i = 1$, alors $g_1 = 0$, donc $\xi < c_1$; et $\omega_1 \geqslant 1$, donc $\xi = \omega \cdot c \geqslant c_1$. Ceci est contradictoire.

Question 32 • $\omega_i \neq 0$, donc $\xi \geqslant c_i$; mais $g_i = 0$, donc il existe un indice $s \in [\![1, i-1]\!]$ tel que $g_s > 0$. Par suite, $\xi \geqslant c_s$, et à plus forte raison $\xi \geqslant c_{i-1}$.

• Notons μ le m-uplet défini par $\mu_j = \omega_j - 1$ et $\mu_k = \omega_k$ pour $k \neq j$. D'après le résultat de la question 29, μ est la représentation optimale de $\xi - c_j$. Comme ξ est le plus petit contre-exemple, $\Omega_c(\xi - c_j) = \Gamma_c(\xi - c_j)$; cette représentation de $\xi - c_j$ n'utilise aucune valeur supérieure à c_i, donc $\xi - c_j < c_{i-1}$. Nous avons ainsi l'encadrement $\xi - c_j < c_{i-1} \leqslant \xi$.

• Notons $\gamma' = \Gamma_c(c_{i-1} - 1)$; comme $c_{i-1} - 1 \geqslant c_i$, il est certain que $\gamma'_i \neq 0$. Nous pouvons donc déduire de γ' et ω, par décrémentation de γ'_i et ω_i, des représentations gloutonnes γ'' de $c_{i-1} - 1 - c_i$ et ω' de $\xi - c_i$. Mais $c_{i-1} - 1 - c_i < \xi - c_i$, donc $\gamma'' \prec \omega'$. En ajoutant 1 à γ''_i et ω'_i, nous ne changeons pas l'ordre lexicographique, si bien que $\gamma' \prec \omega$.

• Par ailleurs, en remplaçant ω_j par $\omega_j - 1$, nous obtenons une représentation gloutonne de $\xi - c_j$. Comme $\xi - c_j \leqslant c_{i-1} - 1$, nous avons $\omega' \leqslant g'$. Ainsi, $\omega' \leqslant g' \leqslant m$. Notons que ω' et ω ne diffèrent que par leurs j-ièmes composantes : donc toute composante de g', d'indice inférieur à j, est égale à la composante de même indice de ω. Ainsi, $g'_k = \omega_k$ pour $1 \leqslant k < j$. Mais $g' \prec \omega$: donc il doit exister un indice $k \geqslant j$ tel que $g'_k < \omega_k$. Comme $\omega_k = 0$ pour $k > j$, la seule possibilité est $k = j$: donc $g'_j < \omega_j$. Mais $\Gamma_c(\xi - c_j) \preceq g'$, donc $\omega_j - 1 \leqslant g'_j$. Conclusion : $\omega_j = g'_j + 1$.

Question 33 • Il existe $\mathcal{O}(m^2)$ couples (i, j) d'indices tels que $1 < i \leqslant j \leqslant m$. Un tel couple étant fixé, on calcule $g(c_{i-1} - 1)$ puis $w = g \cdot c + c_j$; on calcule alors $g(w)$, et on vérifie que cette représentation est égale à la représentation optimale de w, que l'on déduit de g en augmentant g_j de 1. Le coût de ces calculs est un $\mathcal{O}(m)$. Le coût total de l'algorithme est donc un $\mathcal{O}(m^3)$.

Question 34 **À FAIRE !** (programmation de l'algorithme de Pearson)

Annexe : où l'on prouve que le système euro est canonique

• Soit $x \geqslant 2$ et $k = (k_1, \dots, k_8)$ une représentation minimale de x.

• $k_8 \leqslant 1$: sinon $(k_1, \dots, k_7 + 1, k_8 - 2)$ serait une représentation moins lourde.

• $2k_7 + k_8 \leqslant 4$: sinon $2k_7 \geqslant 5 - k_8 \geqslant 4$ donc $k_7 = 2$ et $k_8 \geqslant 1$ mais alors $(k_1, \dots, k_6 + 1, 0, k_8 - 1)$ serait une représentation moins lourde ; ou alors $k_7 \geqslant 3$ mais alors $(k_1, \dots, k_7 + 1, k_7 - 3, k_8 + 1)$ serait une représentation moins lourde.

• $5k_6 + 2k_7 + k_8 \leqslant 9$: sinon $5k_6 \geqslant 10 - (2k_7 + k_8) \geqslant 6$ donc $k_6 \geqslant 2$; alors $(k_1, \dots, k5 + 1, k_6 - 2, k_7, k_8)$ serait une représentation moins lourde.

• $10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 19$: sinon $10k_5 \geqslant 20 - (5k_6 + 2k_7 + k_8) \geqslant 11$ donc $k_5 \geqslant 2$; alors $(k_1, \dots, k_4 + 1, k_5 - 2, k_6, k_7, k_8)$ serait une représentation moins lourde.

• $20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 49$: sinon $20k_4 \geqslant 50 - (10k_5 + 5k_6 + 2k_7 + k_8) \geqslant 31$: ou bien $k_4 \geqslant 3$, alors $(k_1, k_2, k_3 + 1, k_4 - 3, k_5 + 1, k_6, k_7, k_8)$ serait une représentation moins lourde ; ou bien $k_4 = 2$ et $10k_5 + 5k_6 + 2k_7 + k_8 \geqslant 10$ donc, puisque $5k_6 + 2k_7 + k_8 \leqslant 9$, $k_5 \geqslant 1$, alors $(k_1, k_2, k_3 + 1, 0, k_5 - 1, k_6, k_7, k_8)$ serait une représentation moins lourde.

• $50k_3 + 20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 99$: sinon $50k_3 \geqslant 100 - (20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8) \geqslant 51$ donc $k_3 \geqslant 2$; alors $(k_1, k_2 + 1, k_3 - 2, k_4, \dots, k_8)$ serait une représentation moins lourde.

• $100k_2 + 50k_3 + 20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8 \leqslant 199$: sinon $100k_2 \geqslant 200 - (50k_3 + 20k_4 + 10k_5 + 5k_6 + 2k_7 + k_8) \geqslant 101$ donc $k_2 \geqslant 2$; alors $(k_1 + 1, k_2 - 2, k_3, \dots, k_8)$ serait une représentation moins lourde.

• On a quasiment terminé la preuve : en effet, $0 \leqslant x - 200k_1 = kc - 200k_1 \leqslant 199$ donc k_1 est bien le quotient de x par 200. Puis notant $x_1 = x - 200k_1$, $0 \leqslant x_1 - 100k_2 \leqslant 99$ donc k_2 est bien le quotient de x_1 par 100, puis, de la même manière, k_3 est le quotient de $x_2 = x_1 - 100k_2$ par 50, k_4 le quotient de $x_3 = x_2 - 50k_3$ par 20, k_5 le quotient de $x_4 = x_3 - 20k_4$ par 10, k_6 le quotient de $x_5 = x_4 - 10k_5$ par 5, k_7 le quotient de $x_6 = x_5 - 5k_6$ par 2 et enfin $k_8 = x_7 = x_6 - 2k_7$: la représentation k coïncide avec la représentation gloutonne.

Références bibliographiques

► J'ai utilisé les deux articles suivants, que l'on trouvera facilement sur le Web, en particulier sur le site de NEC (`citeseer`).

► *Optimal bounds for the change-making problem*, de Dexter Kozen et Shmuel Zaks. *Theoretical Computer Science*, 123 : 377-288, 1994.

► *A polynomial-time algorithm for the change-making problem*, de David Pearson. Technical Report TR94-1433, Cornell University, Ithaca, NY, June 1994.

FIN