Sujet XENS 2012 : correction

Question 1
Il s'agit de

$$\{\{2\},\{1\},\{0\},\{0,1,2\}\}.$$

Question 2
L'arbre correspondant à $\{\{0\}\}$ est

$$0 \to (\bot, \top)$$

celui correspondant à $\{\emptyset,\{0\}\}$ est

$$0 \to (\top, \top)$$

et celui de $\{\{0,2\}\}$ est

$$0 \to (\bot, 2 \to (\bot, \top))$$

Question 3
Un arbre combinatoire possède une hauteur valant au moins zéro.
Soit A un arbre combinatoire de hauteur zéro différent de $\bot$. C'est donc $\top$; il a donc une feuille $\top$.
Soit $n \in \mathbb{N}$. Supposons que tout arbre combinatoire de hauteur au plus n différent de $\bot$ possède une feuille $\top$. Soit A un arbre combinatoire de hauteur $n+1$. On a $A = i \to (A_1, A_2)$. A_1 et A_2 sont des sous-arbres combinatoires de hauteur au plus n. Par hypothèse, $A_2 \neq \bot$ donc A_2 est un arbre combinatoire différent de $\bot$ de hauteur au plus n. En vertu de l'hypothèse de récurrence, A_2 possède une feuille $\top$ donc A aussi.
Par récurrence, tout arbre combinatoire possède une feuille $\top$.
Question 4
Notons $\mathcal{A}^*(E)$ l'ensemble des arbres combinatoires étiquetés par $E \subset [\![0, n-1]\!]$ différents de $\bot$.
L'ensemble des arbres combinatoires différents de $\bot$ noté $\mathcal{A}^*([\![0,n-1]\!])$ est en bijection avec l'union disjointe des $\{i \,/\, 0 \leq i \leq n-1\} \times (\mathcal{A}^*([\![i+1,n-1]\!]) \cup \{\bot\}) \times \mathcal{A}^*([\![i+1,n-1]\!])$. Ceci découle de la définition inductive donné.
Comme $[\![i+1,n-1]\!]$ est en bijection avec $[\![0,n-i-1-1]\!]$, en notant a_n le cardinal de $\mathcal{A}^*([\![0,n-1]\!])$, on a pour tout $n \geq 1$,

$$a_n = \sum_{i=0}^{n-1} (a_{n-i-1}+1) a_{n-i-1}.$$

Montrons par récurrence que pour tout $n \geq 0$, $a_n = 2^{2^n} - 1$.
Les arbres combinatoires étiquetés par $[\![0,-1]\!]$ sont $\bot$; ainsi $a_0 = 1 = 2^1 - 1 = 2^{2^0} - 1$.
Les arbres combinatoires étiquetés par $[\![0,1-1]\!]$ différents de $\bot$ sont $\top$, $0 \to (\bot, \top)$ et $0 \to (\top, \top)$. Ainsi, $a_1 = 3 = 2^{2^1} - 1$.
Soit $n \geq 1$. Supposons que pour tout $0 \leq k \leq n-1$, $a_k = 2^{2^k} - 1$. Alors, d'après la formule démontrée ci-dessus,

$$a_n = \sum_{i=0}^{n-1} 2^{2^{n-i-1}} \times (2^{2^{n-i-1}} - 1) = \sum_{i=0}^{n-1} 2^{2 \times 2^{n-i-1}} - \sum_{i=0}^{n-1} 2^{2^{n-i-1}}$$

$$= \sum_{i=0}^{n-1} 2^{2^{n-i}} - \sum_{i=0}^{n-1} 2^{2^{n-i-1}}$$

$$= \sum_{i=0}^{n-1} 2^{2^{n-i}} - \sum_{i=1}^{n} 2^{2^{n-i}} = 2^{2^n} - 2^{2^{n-n}} = 2^{2^n} - 1.$$

Ainsi, par récurrence forte, pour tout $n \in \mathbb{N}$, $a_n = 2^{2^n} - 1$. Par conséquent le nombre d'arbres combinatoires étiquetés par $[\![0,n-1]\!]$ est $2^{2^n} - 1 + 1 = 2^{2^n}$ (il y a $\bot$ et ceux différents de $\bot$ qui sont en nombre a_n).
Question 5

```
let rec un_elt a = match a with
   | Un -> []
   | Comb(i,_,a2) -> i::(un_elt a2)
   | _ -> failwith "erreur";;
```

Question 6

```
let rec singleton s = match s with
   | [] -> Un
   | x::q -> Cons x Zero (singleton q);;
```

Question 7

```
let rec appartient s a = match s,a with
   | _,Zero -> false
   | _,Un -> s=[]
   | y::q,Comb(x,a1,a2) when x=y -> appartient q a2
   | y::q,Comb(x,a1,a2) when x<y -> false
   | s,Comb(x,a1,a2) -> appartient s a1;;
```

Question 8

```
let rec cardinal a = match a with
   | Zero -> 0
   | Un -> 1
   | Comb(_,a1,a2) -> (cardinal a1)+(cardinal a2);;
```

Question 9
Il y a 6 sous-arbres, sa taille est 6.
Question 10

```
let cardinal a =
   let t=cree1() in
   let rec aux a = if not present1 t a then match a with
      | Zero -> ajoute1 t a 0
      | Un -> ajoute1 t a 1
      | Comb(_,a1,a2) ->
         aux a1;
         aux a2;
         ajoute1 t a ((trouve1 t a1)+(trouve1 t a2)) end
   in aux a;
   trouve1 t a;;
```

La fonction "aux" sur un arbre a n'utilise que les parties de la table t correspondant à un sous-arbre de a. Pour chacun de ces sous-arbres, un nombre $O(1)$ d'opérations de complexité $O(1)$ est effectuée. On a donc un nombre d'opérations de l'ordre de $O(T(a))$ donc une complexité de $O(T(a))$.
Question 11

```
let inter a1 a2 =
   let t=cree2() in
   let rec aux a1 a2 = if not present2 t a1 a2 then
   match a1,a2 with
      | Zero,_ -> ajoute2 t a1 a2 a1
      | _,Zero -> ajoute2 t a1 a2 a2
      | Un,Un -> ajoute2 t a1 a2 a1
      | Un,Comb(_,a3,_) -> aux a1 a3; ajoute2 t a1 a2 (trouve2 t a1 a3)
      | Comb(_,a3,_),Un -> aux a3 a2; ajoute2 t a1 a2 (trouve2 t a2 a3)
      | Comb(x,a3,a4),Comb(y,a5,a6) when x=y ->
      aux a3 a5; aux a4 a6;
      if trouve2 t a4 a5 = Zero then ajoute2 t a1 a2 (trouve2 t a3 a5)
      else ajoute2 t a1 a2 (cons x (trouve2 t a3 a5) (trouve2 t a4 a6))
      | Comb(x,a3,_),Comb(y,_,_) when x<y -> aux a3 a2; ajoute2 t a1 a2 (trouve2 t a3 a2)
      | Comb(x,_,_),Comb(y,a5,_) -> aux a1 a5; ajoute2 t a1 a2 (trouve2 t a1 a5)
   in aux a1 a2;
   trouve2 t a1 a2;;
```

Question 12
D'après l'algorithme décrit à la question précédente, il existe une application (décrite dans la fonction précédente), $\mathcal{U}(A_1) \times \mathcal{U}(A_2) \to \mathcal{U}(A_1 \cap A_2)$ qui est surjective. On en déduit que $\text{Card}(\mathcal{U}(A_1 \cap A_2)) \leq \text{Card}(\mathcal{U}(A_1) \times \mathcal{U}(A_2))$ donc $T(A_1 \cap A_2) \leq T(A_1)T(A_2)$.
Question 13
On peut le placer de façon "horizontale". Il y a p lignes possibles. Sur chaque ligne, il y a $p-1$ possibilités. On a donc $p(p-1)$ possibilités horizontales. De même, il y a $(p-1)p$ manières de placer ce domino de façon verticale. Au total, il y $2p(p-1)$ façons de placer un domino dans l'échiquier.
Question 14

```
let colonne j =
   let rec aux i b = match i with
      | n -> if b then Un else Zero
      | _ ->
      if not(m.(i).(j)) then cons(i,aux (i+1) b,aux (i+1) b)
      else if not(b) then cons(i,aux (i+1) true, aux (i+1) true)
      else aux (i+1) b
   in aux 0 false;;
```

Question 15

```
let pavage =
   let a= ref colonne 0 in
```

```
   for j = 1 to Array.length m.(0) -1 do
      a:=inter !a (colonne j)
      done;
   !a;;
```

Question 16

```
let ajoute t k v = t.(hache(k)) <- (k,v)::t.(hache(k));;
```

Question 17

```
let present t k =
   let rec aux l k = match l with
      | [] -> false
      | x::q -> egal (fst x) k || aux q k
   in aux t.(hache(k)) k;;
```

Question 18

```
let trouve t k =
   let rec aux l k = match l with
      | y::q -> if egal (fst y) k then snd(y) else aux q k
      | _ -> failwith "erreur"
   in aux t.(hache(k)) k;;
```

Question 19
Il faut que H soit assez grand pour que chaque seau soit de taille bornée (par exemple si H est de l'ordre du nombre de valeurs et que les valeurs sont réparties sont façon homogène dans les seaux). Pour que les seaux aient un nombre semblable de valeurs, il faut aussi que le nombre d'antécédent d'une valeur prise par la fonction `hache` soit de taille semblable.

Question 20
Supposons que `egal`$(A_1) =$`egal`(A_2).
Si A_1 est une feuille $\bot$ ou $\top$, alors, d'après la définition de `egal`, A_2 est aussi une feuille et $A_1 = A_2 = \top$ ou $A_1 = A_2 = \bot$. On a donc `hache`$(A_1) =$`hache`(A_2) (qui vaut 0 ou 1 dans ce cas).
Si $A_1 = (i_1 \to L_1, R_1)$, alors A_2 ne peut être une feuille donc $A_2 = (i_2 \to L_2, R_2)$. Par définition de `egal`, $i_1 = i_2$, `unique`$(L_1) =$`unique`(L_2) et `unique`$(R_1) =$`unique`(R_2). On a donc `hache`$(A_1) = (19^2 \times i_1 + 19 \times$`unique`$(L_1) +$`unique`$(R_1))) \mod H = (19^2 \times i_2 + 19 \times$`unique`$(L_2) +$`unique`$(R_2)) \mod H =$`hache`$(A_2)$.

Question 21

```
let t = cree1();;
let unique = ref 2;;
ajoute t Zero Zero;;
ajoute t Un Un;;
let rec cons i a1 a2 =
   let a= Comb(!unique,i,a1,a2) in
   if present t a then trouve t a
   else begin
           ajoute t a a;
           unique:=!unique+1;
           a
           end;;
```

Question 22
1. On va, selon le calcul de la fonction `hache`, trouver un seau de longueur i comprise entre 0 et 7. Ensuite on utilisera i fois la fonction `egal` pour vérifier si l'arbre n'est pas déjà dans le seau obtenu. En moyenne, on aura un nombre d'appels de `egal` :

$$\frac{6450}{19997} \times 0 + \frac{7340}{19997} \times 1 + \frac{4080}{19997} \times 2 + \frac{1617}{19997} \times 3 + \frac{400}{19997} \times 4 + \frac{96}{19997} \times 5 + \frac{11}{19997} \times 6 + \frac{3}{19997} \times 7$$

$$\approx 1,13$$

2. On tomber dans un seau de longueur i compris entre 1 et 7. On va trouver en moyenne l'arbre cherchée en $(i+1)/2$ appels de fonctions égal. En outre, le nombre d'arbres associés à un seau de taille i est $i \times n_i$ où n_i est le nombre de seaux de longueur i. La probabilité de trouver un seau de taille i est donc $\frac{n_i \times i}{22518}$. Le nombre moyen d'appels de fonction `egal` sera donc

$$\frac{7340}{22518} \times 1 + \frac{2 \times 4080}{22518} \times 1,5 + \frac{3 \times 1617}{22518} \times 2 + \frac{4 \times 400}{22518} \times 2,5 + \frac{5 \times 96}{22518} \times 3 + \frac{6 \times 11}{22518} \times 3,5 + \frac{7 \times 3}{22518} \times 4$$

$$\approx 1,56$$