Mathématiques Discrètes

Devoir surveillé n° 2— le 18 décembre 2018

Prenez le temps de lire ce sujet. Ce devoir comporte 4 exercices. Les exercices sont indépendants. Le barème est sur 22 points. Toute réponse doit être **justifiée**. Calculatrice et documents interdits.

Exercice 1 [3 points]

`''.join(['a','bc', 'd'])` et `''.join(['ab', 'cd'])` sont deux expressions Python produisant la même chaîne `'abcd'`.

Q 1.1 [3 points] Étant donnée une chaîne `s` de longueur n, exprimez en fonction de n le nombre de listes `l`, ne contenant que des chaînes de caractères non vides, telles que `''.join(l) == s`.

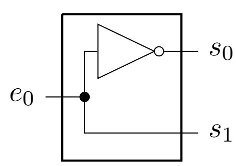
Exercice 2 [5 points]

Un décodeur $(n, 2^n)$ est un circuit combinatoire, ayant n entrées et 2^n sorties. Comme les entrées e_0, e_1, ... ,e_{n-1} prennent leurs valeurs dans l'ensemble $\{0, 1\}$, exactement 2^n valeurs peuvent être prises par les entrées. Chacune peut être représentée par un entier e codé sur n bits.

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

Une seule des sorties s_0, s_1, ... , s_{2^n-1} du décodeur sera à 1, celle qui correspond au nombre e. Toutes les autres sorties sont à 0. Voici la table de vérité du décodeur $(1, 2)$:

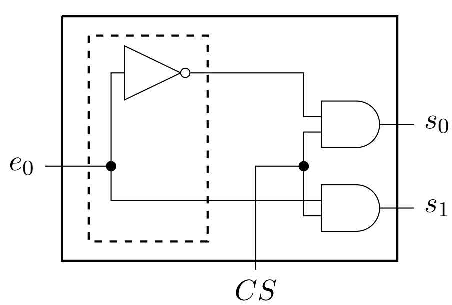
e_0	s_1	s_0
0	0	1
1	1	0



Pour des raisons pratiques, on ajoute une entrée supplémentaire au circuit nommée CS (chip select), cette entrée agit comme un interrupteur marche/arret du circuit. C'est-à-dire, lorsqu'elle est à 0, toutes les sorties du circuit sont mises à 0 et lorsqu'elle est à 1 le circuit fonctionne normalement. Afin de bien préciser voici la table de vérité du circuit :

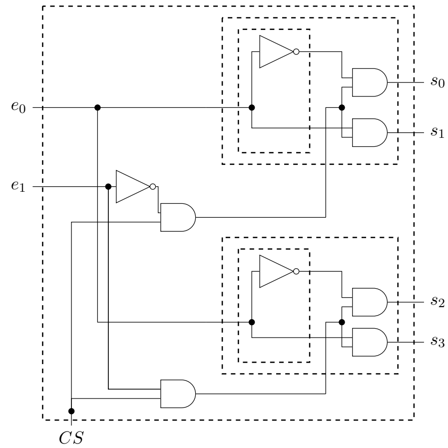
CS	e_0	s_1	s_0
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	0

et une réalisation :

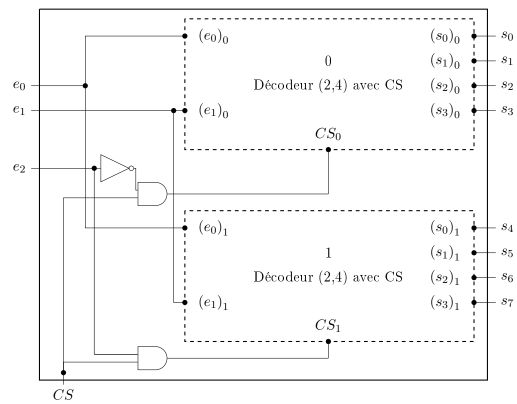


En utilisant deux décodeurs $(1, 2)$ avec CS. On réalise le circuit suivant qui correspond à un décodeur $(2, 4)$ avec CS :

CS	e_1	e_0	s_3	s_2	s_1	s_0
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



En utilisant deux décodeurs $(2,4)$ avec CS. On peut réaliser un circuit suivant qui correspond à



un décodeur $(3,8)$ avec CS.

De manière générale, en utilisant deux décodeurs $(n, 2^n)$ avec CS, on est capable de construire un décodeur $(n+1, 2^{n+1})$ avec CS.

On note u_n le nombre de portes logiques (et, non, ou) utilisée pour réaliser un décodeur $(n, 2^n)$ avec CS.

Q 2.1 [1 point] Donnez les valeurs de u_1, u_2, u_3.

Q 2.2 [1 point] Donnez la relation de récurrence satisfaite par les termes de cette suite pour $n \geq 1$. Précisez la nature de la suite.

Q 2.3 [1 point] Quel sens pourrait-on donner à un décodeur $(0,1)$ avec CS ? Si cela avait un sens, combien de portes logiques cela nécessiterait ?

Q 2.4 [2 points] Donnez une expression de u_n directement en fonction de n, c'est-à-dire ne faisant plus intervenir de termes précédents de la suite.

Exercice 3 [9 points]

On donne le code suivant :

```
def C(n,m):
    assert type(n)==int,n>=0
    assert type(m)==int,m>=0
    if n==0 or m==0:
       return 1
    else:
       return C(n-1,m)+C(n,m-1)+C(n-1,m-1)
```

Q 3.1 [1 point] Calculez $C(i, j)$ pour i et j inferieur ou égal à 3.

Q 3.2 [1 point] Démontrez que pour tout couple d'entier $C(n,m) = C(m,n)$.

On considère les mots sur l'alphabet $\mathcal{A} = \{U, R, D\}$.

On définit pour tout mot m de $\mathcal{A}^*$ respectivement les nombres $|m|_U$,$|m|_R$, et $|m|_D$ comme les nombres d'occurrences respectifs des lettres U, R et D.

Une relation $\mathcal{R}$ sur les mots est définie par $m_1 \mathcal{R} m_2$ si par définition

$$|m_1|_U + |m_1|_D = |m_2|_U + |m_2|_D \text{ et } |m_1|_R + |m_1|_D = |m_2|_R + |m_2|_D.$$

Q 3.3 [1 point] Montrez que $\mathcal{R}$ est une relation d'équivalence.

On interprète chaque mot de $\mathcal{A}^*$ comme un chemin tracé depuis l'origine $(0,0)$ en ne faisant que des pas vers le haut (U) des pas vers la droite (R), et des pas en diagonale (D).

$$U : (x, y) \mapsto (x, y + 1)$$

$$R : (x, y) \mapsto (x + 1, y)$$

$$D : (x, y) \mapsto (x + 1, y + 1).$$

Q 3.4 [1 point] Dessinez le chemin $UUDRDDURRRUD$.

Q 3.5 [1 point] Que dire de deux mots dont l'interprétation mène au même point ? Donnez la signification de la relation $\mathcal{R}$.

Q 3.6 [1 point] Donnez la classe d'équivalence de UR, celle de UUR.

Q 3.7 [1 point] Dans la classe d'équivalence de $UURR$, combien y a t-il de mots sans D ? avec un seul D ? Combien y a-t-il de D au maximum ?

On note $C(n,m)$ le nombre mots dans la classe d'équivalence de $U^n R^m$ où U est répété n fois et R est répété m fois.

Q 3.8 [1 point] Quelle relation de récurrence lie $C(n,m)$ aux nombres $C(n-1,m)$, $C(n-1,m-1)$ et $C(n,m-1)$? Vous justifierez soigneusement.

Dans la suite on définit $Dc(n) = C(n,n)$.

Q 3.9 [1 point] Démontrez que

$$Dc(n) = \sum_{k=0}^{n} \binom{n+k}{k} \binom{n}{k}.$$

Exercice 4 [5 points]

On veut réaliser une fonction `arbroazar` en Python qui fabrique un arbre binaire étiqueté par `None`, de taille n, choisi pseudo-aléatoirement. On souhaite que la probabilité soit uniforme sur l'ensemble des arbres de taille n. C'est-à-dire que chaque arbre de taille n possède exactement la même probabilité d'être fabriqué.

Q 4.1 [1 point] Dessinez un arbre binaire de taille 6. Quelle est la probabilité qu'il soit fabriqué par un appel à la fonction `arbroazar` (supposée réalisée) ?

T.S.V.P.

Une classe `Binary_Tree` a été réalisée en Python. Elle permet de construire soit un arbre vide, soit un arbre binaire dont on donne les deux sous-arbres gauche et droit. Voici une trace d'une session utilisant cette classe :

```
>>> a = Binary_Tree()
>>> a.is_empty()
True
>>> b = Binary_Tree(a,a)
>>> b.is_empty()
False
>>> print(b)
()
>>> c = Binary_Tree(a,b)
>>> print(c)
()()
```

On donne maintenant le code de la fonction f paramétrée par un entier naturel.

```
def f(n):
     """ :CU: n>=0 """
     if n==0:
         return Binary_Tree()
     else:
         k=randint(0,n-1)
         g=f(k)
         d=f(n-1-k)
         return Binary_Tree(g, d)
```

Q 4.2 [1 point] Que vaut $f(n)$?

Q 4.3 [1 point] Dessiner tous les résultats possibles pour $f(3)$. Indiquer leur probabilité d'apparition, en supposant l'indépendance, et l'uniformité des tirages aléatoires fournit par `randint`.

Dans la suite n désigne un entier, et k un entier tel que $k < n$
Q 4.4 [1 point] Combien y a-t-il d'arbres de taille n dont le sous arbre gauche est de taille égale à k. Donnez le code d'une fonction nommee `nb_arbre` paramétrée par n et k qui calcule ce nombre.

Voici le code d'une fonction, la fonction `catalan` attend un entier n en paramètre et renvoie le nombre d'arbres binaires de taille n dont les noeuds contiennent `None` .

```
def determine_indice(n,numero):
    k=0
    cumul =0
    while cumul<=numero:
        cumul+=catalan(k)*catalan(n-1-k)
        k+=1
    return k-1
```

Les contraintes d'utilisation de cette fonction `determine_indice` sont

— `n` est un entier naturel et

— `numero` est un entier naturel inférieur strictement à `catalan(n)`.

Elle permet d'associer à chaque numéro d'arbre binaire de taille n, la taille de son sous-arbre gauche.
Q 4.5 [1 point] En utilisant la fonction `determine_indice`. Proposez une implantation, de la fonction `arbroazar`
