L3 Maths & Info

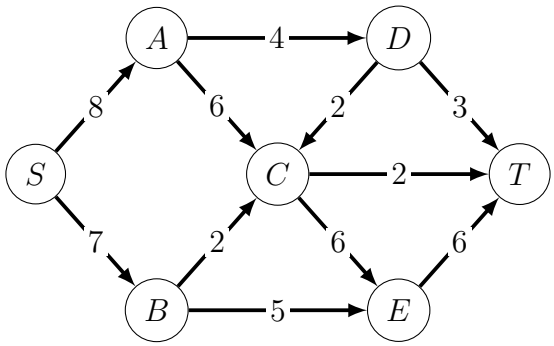
Cours d'Algorithmique 1

Devoir Final - Mardi 14 décembre 2021 - 2h (14h - 16h)

Les notes de cours et de TD sont autorisées. Pour les questions demandant de dérouler un algorithme, vous veillerez à faire apparaître les étapes de calcul permettant de juger de votre bonne compréhension (le résultat final est rarement suffisant). Distinguez bien complexité temporelle et mémoire.

1 Flots (10min)

Question 1. Appliquer l'algorithme de Ford-Fulkerson au réseau de flots ci-dessous. Pour chaque étape, vous donnerez, s'il existe, un chemin améliorant et sa valeur, et le graphe résiduel obtenu après ajout de ce chemin au flot.



Question 2. Identifier la coupe minimale associée au flot trouvé à la question précédente et justifier de la terminaison de l'algorithme.

2 Codages de Huffman (30min)

Il existe de nombreuses façons pour représenter les données. Couramment, on représente les données sous la forme d'un codage binaire des caractères, que l'on abrégera en **codage**, dans lequel chaque caractère est représenté par une chaîne de bits unique.

L'encodage d'un caractère dans un codage est appelé un **code**. On notera code(c) le code associé au caractère $c \in C$.

On distinguera les **codages de longueur fixe** pour lesquels chaque caractère est encodé par une séquence binaire de même longueur. Par exemple, on peut utiliser 3 bits pour représenter six caractères : `a = 000`, `b = 001`, ..., `f = 101` ; et les **codages de longueur variable** pour lesquels le nombre de bits associés à chaque caractère dépend de sa fréquence : un caractère très fréquent (resp. peu fréquent) aura une séquence courte (resp. longue).

C	a	b	c	d	e
$f(c)$	7	45	2	36	78

TABLE 1 – Exemple d'alphabet C avec fréquences f.

C	a	b	c	d	e
Longueur fixe	000	001	010	011	100
Longueur variable	1100	10	1101	111	0

TABLE 2 – Exemple de codages de taille fixe et variable.

Un codage est dit **préfixe** si aucun mot de code n'est préfixe d'un autre, *i.e.* pour tout mot de code u, il n'existe aucun mot de code v tel que : $v = u \cdot w$ où w est une séquence de bits et $\ll \cdot \gg$ désigne la concaténation.

Tout codage préfixe peut être représenté sous la forme d'une arborescence binaire T tel que :

— une feuille est un couple composé d'un caractère $c \in C$ et de sa fréquence $f(c)$;

— un nœud interne est la somme des fréquences des feuilles du sous-arbre dont il est la racine ;

Les deux codages présentés en Table. 2 sont préfixes. Leur arborescences sont présentées ci-dessous (Fig. 1 et Fig. 2).

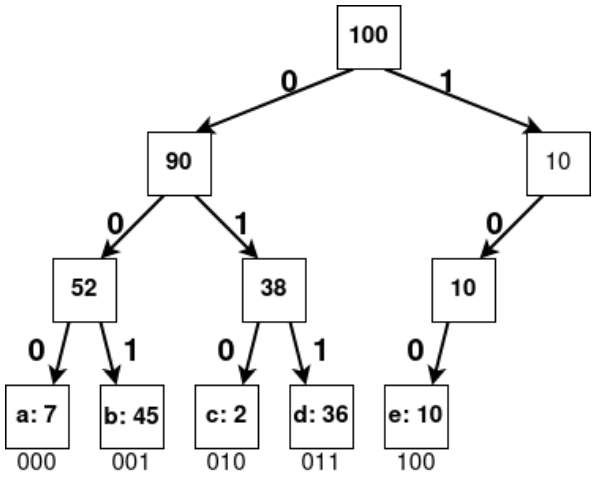


FIGURE 1 – Arborescence du *codage de longueur fixe* du tableau Table. 2

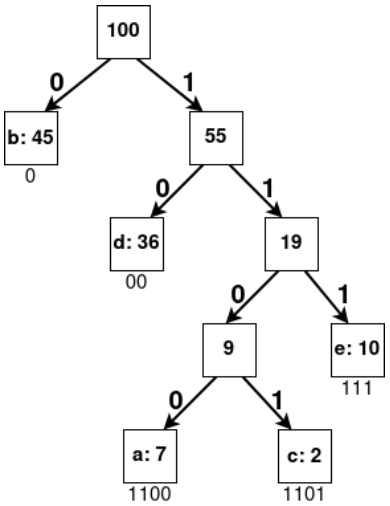


FIGURE 2 – Arborescence du *codage de longueur variable* du tableau Table. 2

Le code associé à un caractère est interprété comme le chemin allant de la racine vers ce caractère avec : 0 si on descend sur l'enfant gauche d'un nœud et 1 si on descend sur l'enfant droit. Le coût d'un code pour un fichier est le nombre total de bits nécessaires pour coder ce fichier :

$$\text{coût} = \sum_{c \in C} f(c) \times \text{code}(c).$$

Un codage est dit **optimal** s'il n'existe aucun autre codage ayant un coût strictement plus petit pour encoder un fichier. (le coût du codage fixe (Fig. 1) est de $\text{coût}_{\text{fixe}} = 300$, et celui du codage variable (Fig. 2) est $\text{coût}_{\text{variable}} = 183$). Le codage de taille variable présenté ci-dessus est optimal.

Dans cet exercice, nous allons nous intéresser au **codage de Huffman** : un codage préfixe optimal de longueur variable.

Question 1. Donner, sans justifier, un codage préfixe optimal et son arborescence T pour l'alphabet $C = \{a, b, c, d, e, f\}$ de fréquences $f(c)$ définies ci-dessous (Table 3).

Indice : le codage optimal a un coût de 224.

C	a	b	c	d	e	f
$f(c)$	45	13	12	16	9	5

TABLE 3 – Fréquence d'apparition des caractères.

Pour la suite, nous poserons C un alphabet donné, avec une fréquence $f(c)$ définie pour chaque caractère $c \in C$. Soit x et y deux caractères appartenant à l'alphabet C ayant les fréquences les plus basses.

Question 2. Montrer que quelque soit le codage préfixe optimal, x et y ont des mots de code de même longueur.

Question 3. Montrer qu'il existe un codage préfixe optimal pour C dans lequel les mots de code pour x et y ne diffèrent que par le dernier bit.

Soient x et y deux caractères de l'alphabet C ayant les fréquences minimales. Notons C' l'alphabet C privé de $\{x, y\}$ et complété par le caractère z ($C' = (C \setminus \{x, y\}) \cup \{z\}$). La fréquence f pour C' est définie comme pour C avec $f(z) = f(x) + f(y)$.

Question 4. Soit T' une arborescence représentant un code préfixe optimal pour l'alphabet C'. Montrer que l'arborescence T, obtenue à partir de T' en remplaçant le nœud feuille associé à z par un nœud interne ayant x et y comme enfants, représente un code préfixe optimal pour l'alphabet C.

Question 5. En déduire et donner un algorithme glouton renvoyant un codage de Huffman pour un alphabet C et une fréquence $f(c)$ définie pour chaque caractère $c \in C$. Donner sa complexité. *Attention, vous veillerez à donner les complexités des structures de données utilisées par votre algorithme.*

3 Plus longue sous-séquence palindromique (30min)

Notations : pour un mot u, on notera $|u|$ sa longueur et $u[0], u[1], ..., u[|u|-1]$ les lettres qui le composent. On pourra également se référer à ses facteurs grâce à la notation $u[i..j[:= u[i], u[i+1], ..., u[j-1]$. Par exemple, si $u =$*BONJOUR*, on a $u[3] =$*j*, et $u[1..3[=$*ON*. On appelle sous-séquence de u, tout mot v, tel qu'il existe $a_0 < a_1 < ... < a_{|v|-1}$ tels que $v[i] = u[a_i]$, pour tout $0 \leq i \leq |v| - 1$. Ainsi, *BNUR* est une sous-séquence de *BONJOUR*.

On appelle *palindrome* un mot dont l'ordre des lettres reste le même qu'on les lise de gauche à droite ou de droite à gauche. Par exemple les mots A, BOB, ANNA, KAYAK, et LOLILOL sont des palindromes.

On s'intéresse au problème de la plus longue sous-séquence palindromique, i.e. de trouver une sous-séquence de longueur maximale qui soit un palindrome. Par exemple, le mot ***IND****ICEPR****O****GR****AMMA****T****I****ON****D****Y****N****AM****I****QUE* contient la sous-séquence palindromique *INDOAMMAODNI* de longueur maximale 12. Notez que cette solution n'est pas unique, par exemple, *INDIAMMAIDNI* en est également une sous-séquence palindromique de longueur 12.

Question 1. Donner, sans justifier, une plus longue sous-séquence palindromique du mot ENSRENNES

On se donne la fonction suivante :

```
procedure MYSTERE(u, i, j)                    ▷ u est un mot, i et j des indices
    k ← i
    while k ≤ j and u[k] ≠ u[j] do
        k ← k + 1
    return k
```

Question 2. a) Que fait la fonction MYSTERE ?
b) Quelle est sa complexité ?

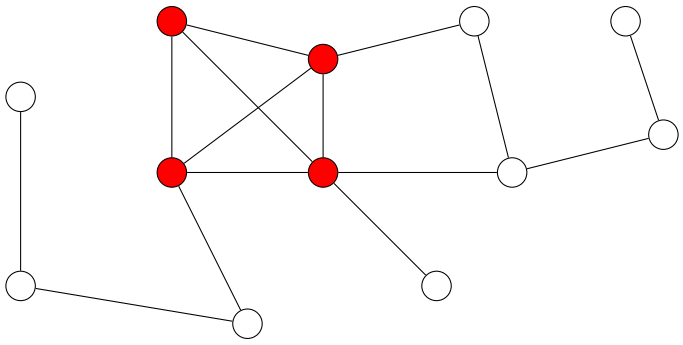
On note $c(u, i, j)$ la longueur d'une plus longue sous-séquence palindromique du facteur $u[i..j[$.

Question 3. On note k le résultat de MYSTERE(u, i, j). Donner, en justifiant, une relation entre $c(u, i, j)$, $c(u, i, j-1)$, et $c(u, k+1, j-1)$.

Question 4. Écrire une fonction permettant de calculer la longueur d'une plus longue sous-séquence palindromique d'un mot u.
Bonus : Vous marquez plus de points si votre fonction permet également de construire une plus longue sous-séquence palindromique.

4 Clique (15min)

Soit $G = (S, A)$ un graphe non orienté. Une clique de taille k dans G est un ensemble de sommets qui sont tous reliés entre eux. Par exemple, dans le graphe suivant, on a marqué les sommets d'une clique de taille 4 :



Question 1. Donner une définition formelle d'une clique de taille k.

Le problème de la clique est défini par :
— entrée : un graphe non orienté G, un entier k écrit en unaire ;
— sortie : oui, si G admet une clique de taille k ; non sinon.

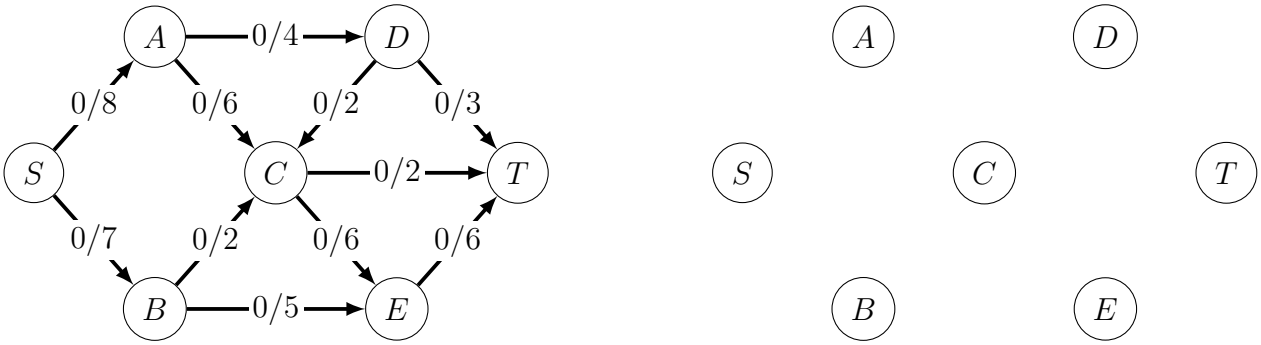
Question 2. Montrer que le problème de la clique est dans NP.

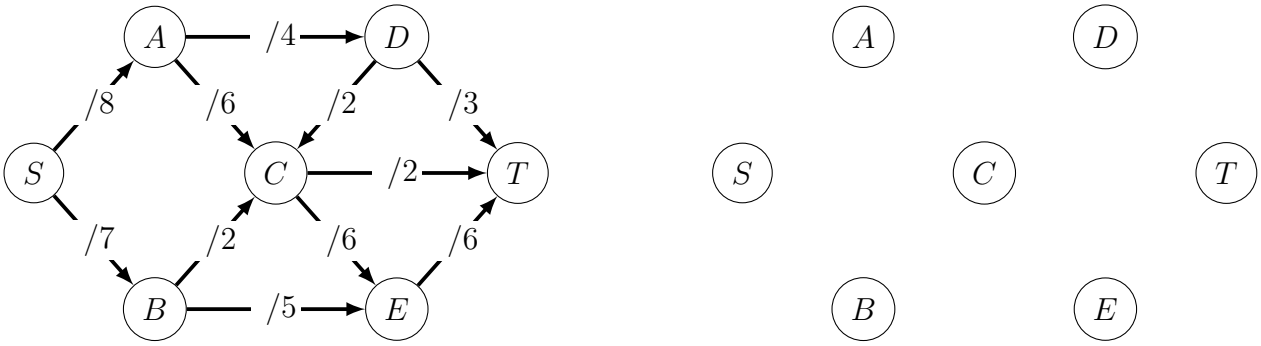
Question 3. Montrer que le problème de la clique est NP-difficile.

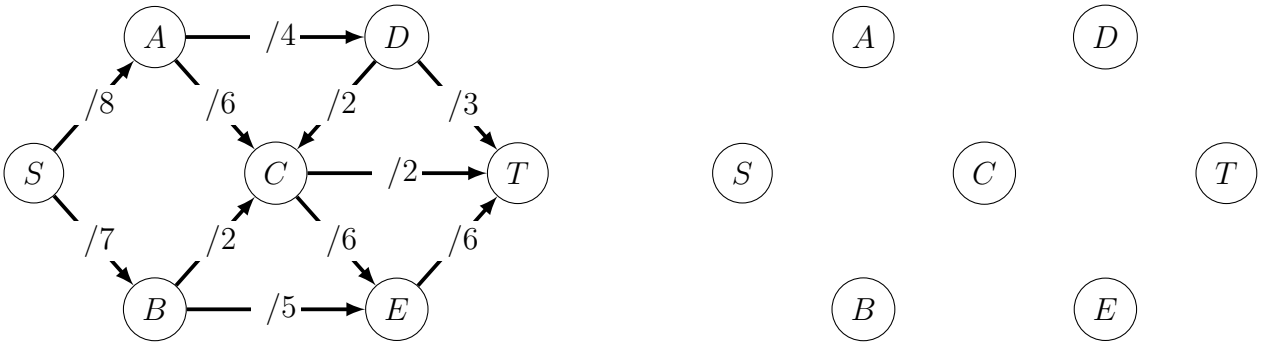
Indice : utiliser le problème des ensembles indépendants.

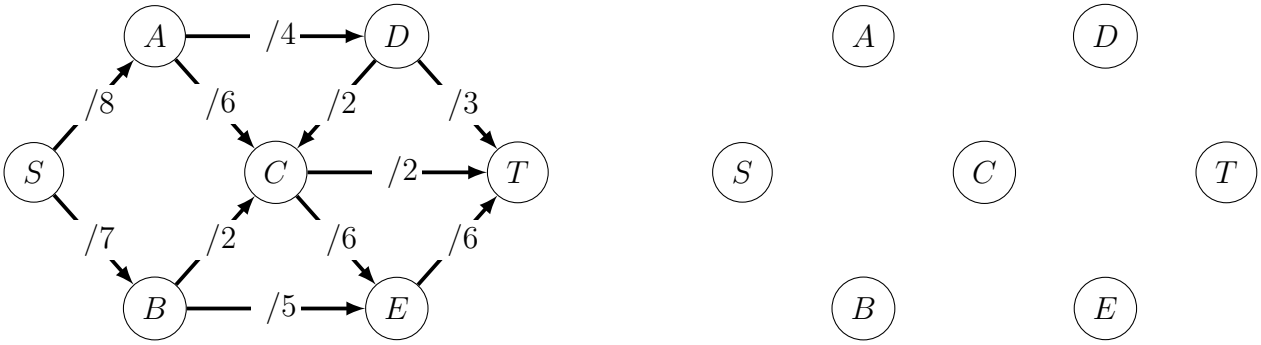
Annexe : exécution de l'algorithme de Ford-Fulkerson

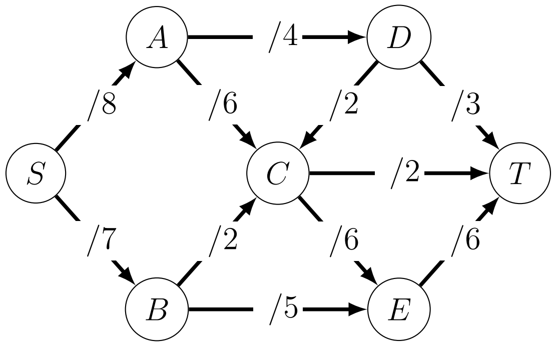
Réseaux Graphes résiduels

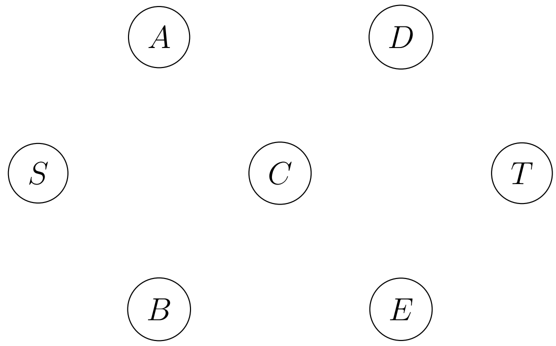


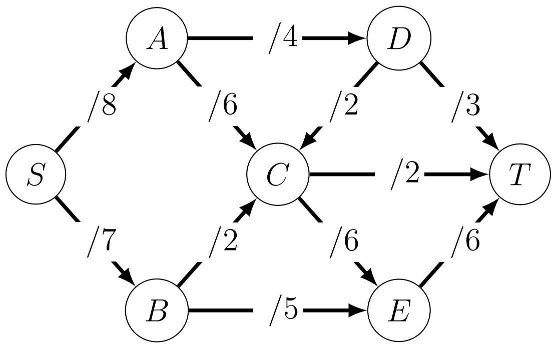


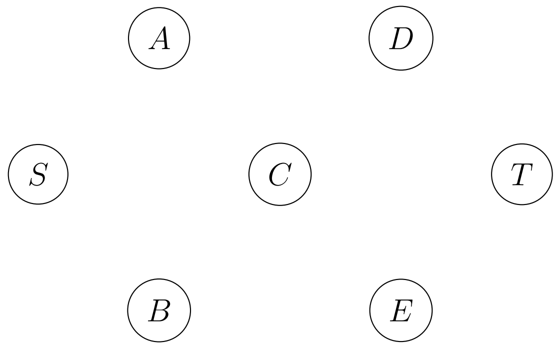


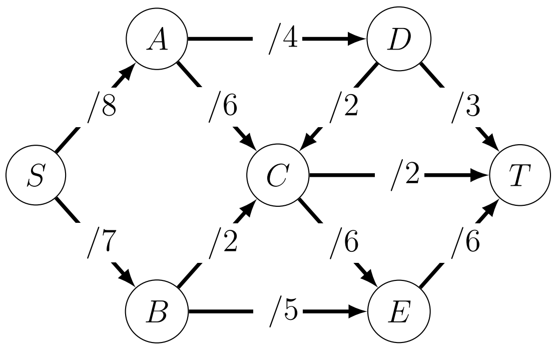


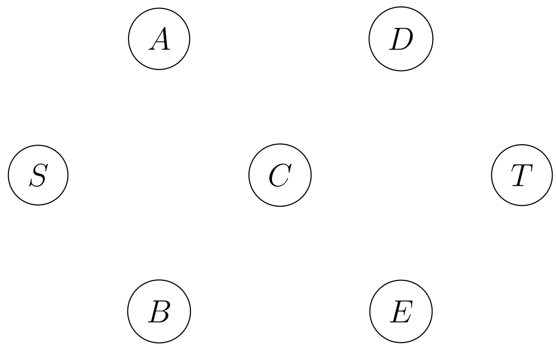

A
D
S
C
T
B
E
/4
/8
/6
/2
/3
/2
/7
/2
/6
/6
/5


A
D
S
C
T
B
E


A
D
S
C
T
B
E
/4
/8
/6
/2
/3
/2
/7
/2
/6
/6
/5


A
D
S
C
T
B
E


A
D
S
C
T
B
E
/4
/8
/6
/2
/3
/2
/7
/2
/6
/6
/5


A
D
S
C
T
B
E