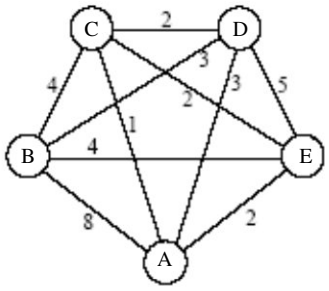
UNIVERSITÉ D'ORLÉANS **M1 Info**

Informatique Année **2018 - 2019**

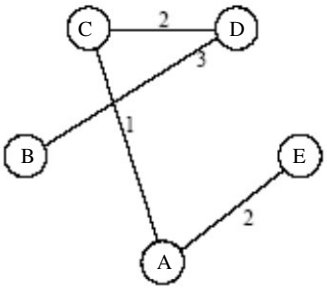
Série de Travaux Dirigés : **4** *- MPI - Parallélisation de l'algorithme de Prim*

Exercice 1. Calcul d'un arbre couvrant minimum

L'objectif de cet exercice est de paralléliser l'algorithme de Prim qui permet de calculer l'arbre couvrant minimal d'un graphe non orienté comme illustré ci-dessous où la matrice indiquée est la matrice d'adjacence du graphe initial:



Undirected graph



Minimum spanning tree

0	8	1	3	2
8	0	4	3	4
1	4	0	2	2
3	3	2	0	5
2	4	2	5	0

L'algorithme séquentiel que vous avez déjà vu est le suivant :

Algorithme ACM-Prim($G = (V, E)$, w, r)

```
F ← V
pour chaque u ∈ F faire
    cle[u] ← ∞
cle[r] ← 0
π[r] ← NIL
tant que F ≠ ∅ faire
    u ← ExtraireMin(F)
    pour chaque v ∈ Adj[u] faire
        si v ∈ F et w({u,v}) < cle[v] alors
            π[v] ← u
            cle[v] ← w({u,v})
```

Ainsi, à partir d'une matrice d'adjacence pour représenter le graphe, il est possible de paralléliser cet algorithme en partageant cette matrice par lignes sur les différents processeurs. Proposez et implémentez une version parallèle de l'algorithme Prim en utilisant les fonctions de communication MPI vues en cours.

Pour créer un graphe, il est proposé de générer une matrice d'adjacence aléatoire sur un processeur et de la distribuer ensuite sur les autres processeurs. Un squelette est disponible sous Celene qui vous permet de faire cette génération et surtout d'écrire le graphe généré au format .dot. Avec la commande *dot -Tpdf nom.dot -o nom.pdf* vous pourrez générer une version .pdf du graphe et donc le visualiser. L'exécution de ce squelette se fait de la manière suivante

mpirun -np ?? ./main arg1 arg2 arg3 arg4

où arg1 est le nombre de nœuds du graphe, arg2 le sommet racine, arg3 le nom du fichier (.dot) de sauvegarde du graphe généré et arg4 le nom du fichier (.dot) du résultat calculé.

Enfin, pour vous aider à vérifier vos résultats, une version séquentielle en python est disponible sous Celene. L'exécution du script se fait de la manière suivante

python3 primMST.py arg1 arg2 > arg3

où arg1 est le fichier .dot du graphe à lire, arg2 le sommet racine et arg3 le fichier (redirection) du résultat.