



# **RAPPORT PROJET-RT803**

**Professeur encadrant : THIBAUT Bernard**

**Etudiants :**

Nadjial Stéphane MBANGOSSOUM

**Année académique: 2019-2020**

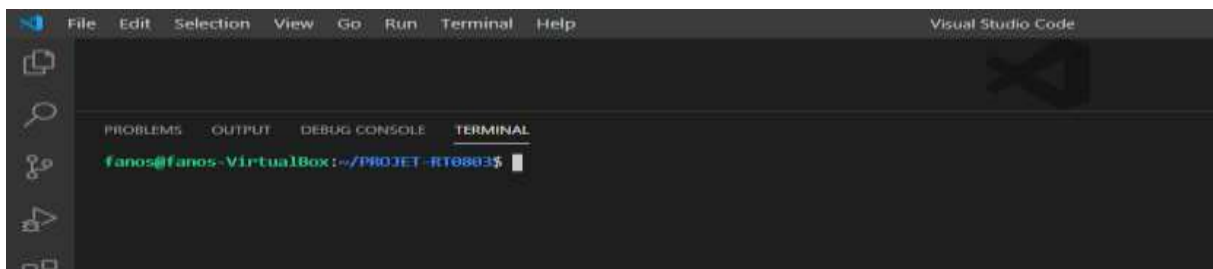
## PLAN

- Algorithme retenu
- Modélisation des données
- Organisation des fichiers
- Méthode d'exécution

Notre projet consiste à réaliser une simulation d'un algorithme distribué, le simulateur doit pouvoir prendre en paramètre un graphe (pondéré ou non, suivant le choix de l'algorithme) sous forme de fichier descriptif (Matrice/Liste d'adjacence ou format plus complexe, au choix).

### - Algorithme retenu

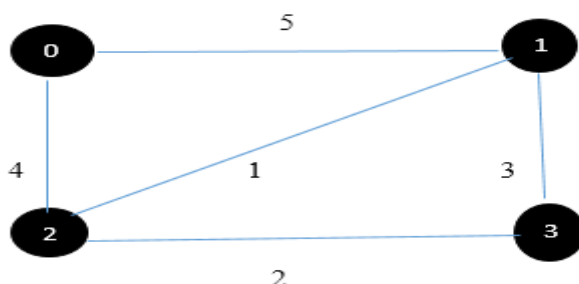
Nous avons choisi de programmer L'algorithme Gallager Humblet Spira(GHS), en langage C++ sur une distribution Linux connectée à VScode.



### - Modélisation des données

L'algorithme prend les données d'un graphe au format d'un fichier sous forme Matrice d'adjacent. L'exemple est basé sur un graphe que nous avons fait pendant les cours en présentiels pour modéliser.

#### Graphe pondère de taille 4

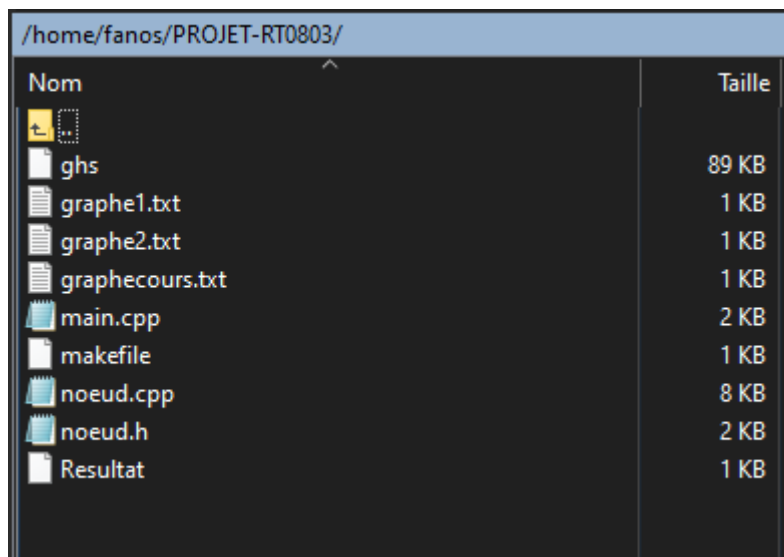


### Matrice associée au graphe,

|   | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 5 | 4 | 0 |
| 1 | 5 | 0 | 1 | 3 |
| 2 | 4 | 1 | 0 | 2 |
| 3 | 0 | 3 | 2 | 0 |

### - Organisation des fichiers

Le projet est organisé sur plusieurs fichiers :



The screenshot shows a file manager window with the path `/home/fanos/PROJET-RT0803/`. It displays a list of files and folders with their names and sizes.

| Nom             | Taille |
|-----------------|--------|
| ghs             | 89 KB  |
| graphe1.txt     | 1 KB   |
| graphe2.txt     | 1 KB   |
| graphecours.txt | 1 KB   |
| main.cpp        | 2 KB   |
| makefile        | 1 KB   |
| noeud.cpp       | 8 KB   |
| noeud.h         | 2 KB   |
| Resultat        | 1 KB   |

**noeud.h** : est un fichier entête qui contient la déclaration de l'ensemble de nos fonctions, nos variables ainsi que nos classes :

- **Classe noeud** : qui définit chaque noeud par rapport au poids de son voisin de gauche et de droite.
- **La fonction MinCanal**: permet de trouver le noeud qui a le poids le plus petit.  
Nous avons mis : valeur de voisin : basic=0, branch=1 et reject =2  
Etat : sleep=0, finf=1 et found=2

**noeud.cpp** : implémentation de nos fonctions déclarées, le cœur du programme

**main.c** : fichier d'exécution du programme, vérification du fichier au paramètre,

**makefile** : fichier de compilation du programmes

**ghs** : fichier exécutable

**Résultat** : fichier de résultat du déroulement

Le déroulement de notre algorithme commence dans la fonction main qui permet de vérifier si un fichier donné en paramètre et respecte le format, il cherche le poids minimum pour ensuite passer au bloc1 initialisation de l'algorithme de GHS, le déroulement se passera sur toute les blocs et retourné le résultat dans fichier définit dans le noeud.cpp

### - Méthode d'exécution

L'exécution du programme se fait de la manière suivante, il faut dans un premier temps mettre tous les fichiers dans un dossier et le compiler avec la commande **make**.

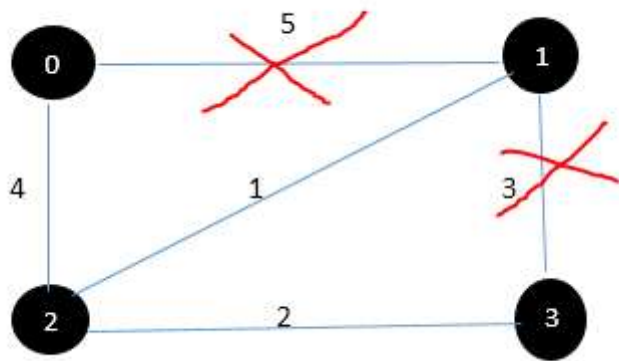
```
fanos@fanos-VirtualBox:~/PROJET-RT0803$ make
g++ -c -o noeud.o noeud.cpp
g++ -pthread -c main.cpp
g++ -pthread -o ghs noeud.o main.o
fanos@fanos-VirtualBox:~/PROJET-RT0803$
```

Comme défini dans le makefile, il crée un fichier ghs, il suffit de lancer programme avec la commande suite **./ghs graphecours.txt**(le fichier en paramètre)

```
fanos@fanos-VirtualBox:~/PROJET-RT0803$ ./ghs graphecours.txt
fanos@fanos-VirtualBox:~/PROJET-RT0803$
```

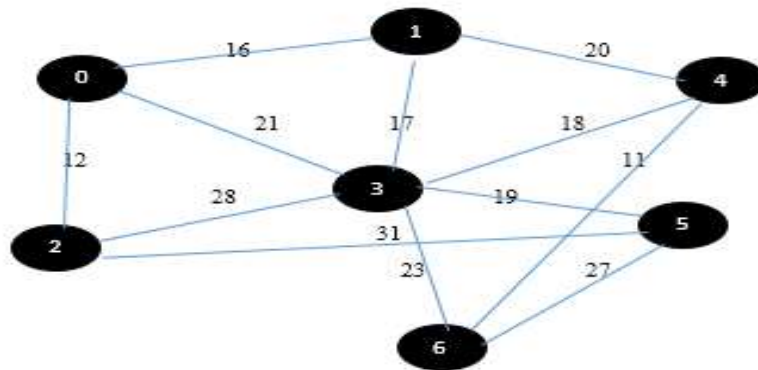
Le résultat est envoyer dans le fichier Resultat,il suffit de faire **cat Resultat** pour avoir les chemin de notre ghs .

```
fanos@fanos-VirtualBox:~/PROJET-RT0803$ cat Resultat
0->2
1->2
2->3
fanos@fanos-VirtualBox:~/PROJET-RT0803$
```



Exemple 2 : Un graphe avec 7 nœuds.

D



Matrice :

```

4-70
fanos@fanos-VirtualBox:~/PROJET-RT0803$ cat graphe1.txt
7
0 16 12 21 0 0 0
16 0 0 17 20 0 0
12 0 0 28 0 31 0
21 17 28 0 18 19 23
0 20 0 18 0 0 11
0 0 31 19 0 0 27
0 0 0 23 11 27 0
fanos@fanos-VirtualBox:~/PROJET-RT0803$

```

Resultat :

```
fanos@fanos-VirtualBox:~/PROJET-RT0803$ cat Resultat
0->1
0->2
1->3
3->5
3->4
4->6
```

