

Université / U.F.R. des Sciences et Techniques

Compte-Rendu de TP Réseau

TP4 : Le Routage

Auteur :

REDJRADJ YACINE

Année Universitaire 2025-2026

Table des matières

1	Rappel du sujet	2
2	Analyse du sujet	2
3	Choix techniques effectués	2
4	Résultats et tests	2
4.1	Représentation visuelle du réseau	2
4.2	Table de routage	3
5	Conclusion	4

1 Rappel du sujet

Il s'agit d'implanter un programme JAVA permettant de donner le chemin le plus court dans une topologie réseau donnée. Le programme doit remplir les fonctions suivantes :

- Permettre de saisir le graphe correspondant à la topologie réseau.
- L'utilisateur choisira deux machines dans le réseau puis le programme permettra de trouver l'ensemble des nœuds intermédiaires ainsi que les interfaces utilisées pour se rendre d'une extrémité à l'autre.
- Permettre d'établir les tables de routage au niveau de chaque commutateur.

2 Analyse du sujet

Le routage consiste à déterminer le chemin optimal pour acheminer des données à travers un réseau de commutateurs. Pour ce faire, le réseau est modélisé sous la forme d'un graphe pondéré où chaque sommet représente un commutateur et chaque arête représente une liaison avec un coût (ou poids) associé. La détermination de la table de routage implique de calculer le chemin de coût minimum vers toutes les destinations possibles, puis de conserver uniquement le "prochain saut" (Next Hop) pour chaque destination.

3 Choix techniques effectués

Pour répondre aux exigences et expliquer la démarche suivie pour la conception des programmes, l'architecture suivante a été mise en place :

- **Librairie GraphStream** : Conformément aux recommandations, la librairie GraphStream a été utilisée pour la représentation des graphes. Elle permet la lecture de topologies depuis un fichier `.dgs` et offre un rendu visuel intégré (`J2DGraphRenderer`).
- **Algorithme de Dijkstra** : Pour mettre en œuvre la fonction qui calcule le plus court chemin entre deux commutateurs, l'algorithme de Dijkstra fourni par `org.graphstream.algorithm.Dijkstra` a été utilisé.
- **Interface Graphique (Swing)** : L'interface permet le chargement dynamique des fichiers de topologie, la visualisation du graphe et la sélection d'un nœud via une `JComboBox` pour afficher sa table de routage spécifique.

4 Résultats et tests

Les tests ont été effectués en chargeant le fichier `graohTD.dgs` contenant une topologie de 6 commutateurs (C1 à C6).

4.1 Représentation visuelle du réseau

Une fois le fichier chargé, le programme génère le rendu visuel du graphe avec les poids associés à chaque liaison (Figure 1).

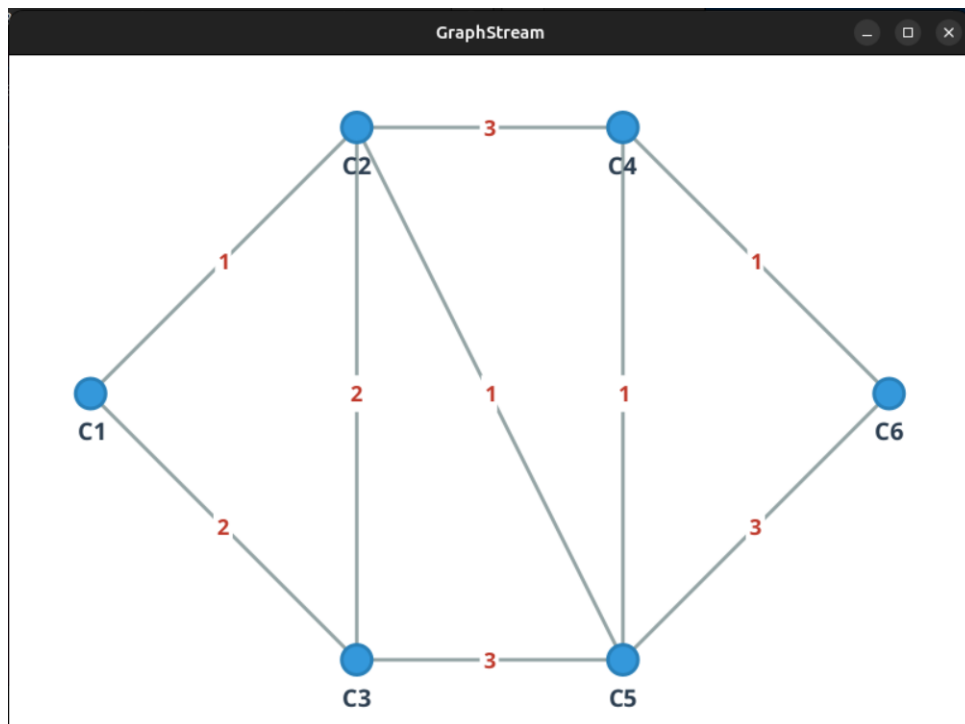


FIGURE 1 – Visualisation de la topologie du réseau avec GraphStream

4.2 Table de routage

En sélectionnant le nœud C1, le programme calcule les plus courts chemins vers tous les autres nœuds (C2, C3, C4, C5, C6) et affiche la table de routage correspondante indiquant le prochain saut et le coût cumulé (Figure 2).

Routing Table Generator

Routing Tables Displayer

Charger Afficher

Chargé : graohTD.dgs

C1 ▼ Afficher Table

Table de routage		
C1	C2 (1)	C3 (2)
C2	C2 (N/A)	C3 (2)
C3	C3 (N/A)	C2 (2)
C4	C2 (3)	C3 (N/A)
C5	C2 (1)	C3 (3)
C6	C2 (N/A)	C3 (N/A)

FIGURE 2 – Table de routage générée pour le commutateur C1

5 Conclusion

L'utilisation conjointe de Java Swing et de GraphStream a permis de concevoir une application modulaire et visuellement intuitive pour simuler le routage réseau.

Difficultés rencontrées : L'intégration de la librairie externe GraphStream a nécessité une gestion rigoureuse du *classpath* lors de la compilation et de l'exécution en ligne de commande. De plus, le tri des voisins pour le formatage correct de la table de routage a demandé une adaptation spécifique du comparateur.

Perspectives : Le programme pourrait être étendu pour calculer et afficher dynamiquement l'établissement d'une ligne de table de circuit virtuel. Une autre évolution consisterait à animer le parcours des paquets directement sur le rendu visuel du graphe.