COMPTE RENDU TRAVAUX PRATIQUES

GRAPHES

SERIE N°2

Optimisation du coût d'un réseau de fibre optique

A l'attention de :

M.OURIACHI Khadir

Réalisé Par :

OLAZAGAZTI Lucas

SERRANO Pierre

1 INTRODUCTION

Le problème abordé ici est soulevé dans le cadre d'un projet de déploiement, par un opérateur, d'un réseau de fibre optique sur une zone uniformément urbanisée. Le problème consiste à optimiser le coût global de déploiement.
Pour résoudre ce problème, nous allons le ramener à un problème de graphe. Pour ce faire, nous allons commencer par modéliser le réseau de fibre optique à l'aide d'un graphe orienté G = (S, A).
Avec :

- S = L'ensemble des nœuds du réseau

  Un nœud Ni = Un sommet Si

- A = L'ensemble des communication entre nœuds du réseau ù

  Une liaison du réseau Ni=>Nj= arc de G (Si,Sj)

1.1 RÉALISATION

Ce problème revient à un problème classique de recouvrement minimal d'un graphe. Le but étant de retrouver un chemin passant par tous les sommets (chaîne hamiltonien) ayant un coût minimal. Pour résoudre ce problème, nous allons utiliser un algorithme de recherche d'arbre couvrant minimum (ACM) :

- Algorithme de Prim
- Algorithme de Kruskal

1.2 CHOIX DE L'ALGORITHME

L'essentiel du problème consiste à choisir entre les différentes algorithme ACM.

- L'algorithme de Prim est un algorithme permettant la recherche d'arbre couvrant minimal de complexité dépendant du nombre de sommet n.

  O(n2)

- L'algorithme de Kruskal est un algorithme permettant la recherche d'arbre couvrant minimal de complexité dépendant du nombre d'arrête m.

  O (m log (m))

Pour faire notre choix, nous devons calculer la densité.

La densité d'un graphe correspond au rapport du nombre d'arêtes sur le nombre total d'arêtes possibles.

La densité d'un graphe peut être approximé par la formule suivante :

$$D \approx m / n^2$$

Avec :

- D la densité
- m le nombre d'arcs/arêtes
- n le nombre de sommets

Si la densité du graphe tend vers 1, alors m tend vers n^2. Donc la complexité de l'algorithme de Kruskal devient en O (n^2 log (n)). Dans ce cas, le choix de l'algorithme sera celui de Prim qui deviens plus efficace que l'algorithme de Kruskal.

Si la densité du graphe tend vers 0, alors $m < n^2$. Donc l'algorithme de Kruskal sera plus efficace que l'algorithme de Prim.

2 ETUDE DE CAS

Soit le graphe G = (S, A), un graphe orienté.

- Avec S = {S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15} et
- A = {(s1,s2), (s1,s4), (s1,s10), (s1,s11), (s1,s15), (s2,s3), (s2,s6), (s2,s9), (s2,s14), (s3,s4), (s3,s7), (s4,s5), (s4,s9), (s4,s15), (s5,s6), (s5,s8), (s5,s12), (s6,s9), (s6,s15), (s7,s6), (s7,s15), (s8,s6), (s8,s7), (s8,s11), (s9,s7), (s9,s8), (s9,s10), (s10,s13), (s11,s10), (s11,s15), (s12,s11), (s13,s12), (s14,s11), (s14,s12), (s15,s10), (s15,s14)}

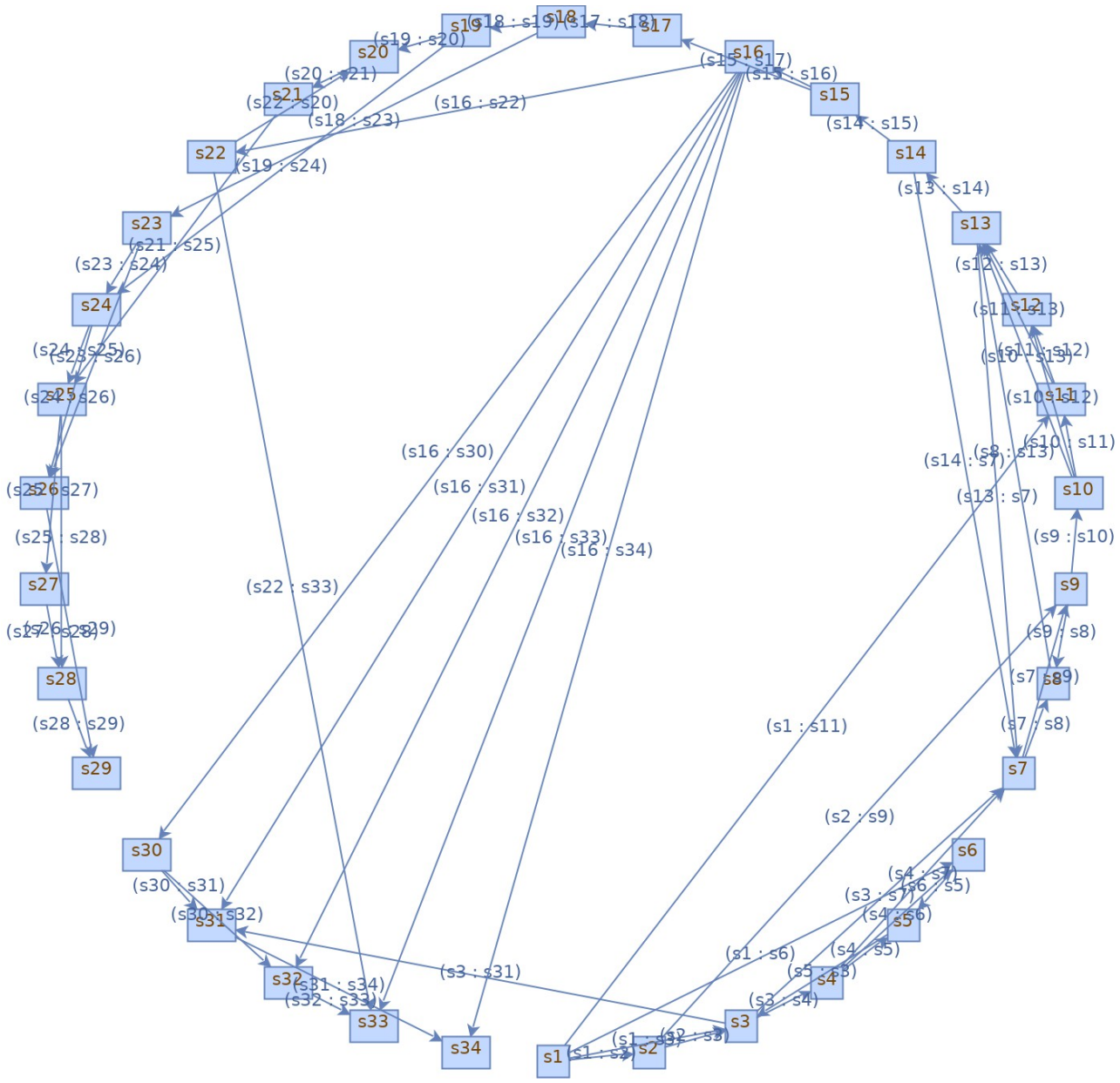

Figure 1: Graphe de l'étude de cas

Le graphe de l'étude de cas possède une densité à peu près égal à 0.0510. Nous allons donc utiliser l'algorithme de Kruskal pour résoudre ce problème.

Soit A =(S, T) l'arbre couvrant du graphe de l'étude de cas.

Avec S l'ensemble des sommets.

S = {s1, s2, s3, s4, s5, s6, s7, s8, s9, s10, s11, s12, s13, s14, s15, s16, s17, s18, s19, s20, s21, s22, s23, s24, s25, s26, s27, s28, s29, s30, s31, s32, s33, s34}

Et T l'ensemble des arcs.

T = {(s1,s2), (s1,s11), (s2,s3), (s4,s5), (s4,s6), (s5,s3), (s7,s8), (s7,s9), (s8,s13), (s10,s11), (s10,s13), (s12,s13), (s14,s7), (s14,s15), (s15,s17), (s16,s22), (s16,s30), (s17,s18), (s18,s19), (s18,s23), (s19,s20), (s20,s21), (s21,s25), (s22,s20), (s22,s33), (s23,s24), (s24,s26), (s25,s27), (s27,s28), (s28,s29), (s30,s31), (s31,s34), (s32,s33)]}

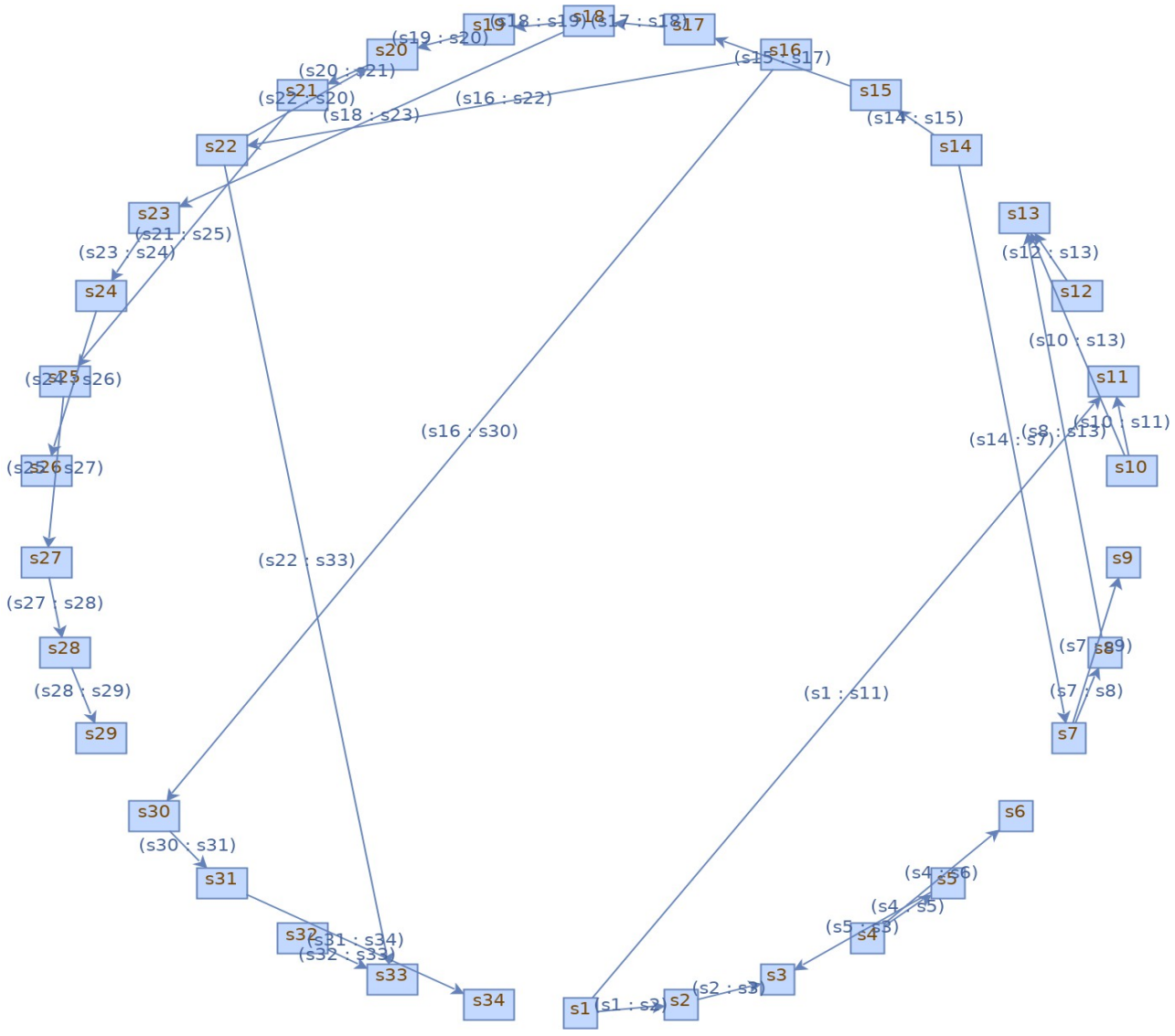

Figure 2: Arbre couvrant du graphe de l'EDC

Après observation de l'arbre couvrant, il existe plusieurs solutions pour le parcourir donc plusieurs solutions optimales à notre étude de cas.

Ces solutions varient en fonction de certains nœuds/sommets: 3, 11, 13, 20, 33.

Le nœud 3 peut être parcouru soit par la branche commençant par le sommet 1 soit par celle commençant par le sommet 5.

Pour le noeud 11 il peut être parcouru soit par la branche commençant par le sommet 1 soit par celle commençant par le sommet 10.

Celui du 13 peut être parcouru par la branche commençant par le sommet 10, soit par celle commençant par le sommet 12 ou par la branche commençant par le sommet 14.

Celui du 33 peut être parcouru par la branche commençant par le sommet 32 ou celle commençant par le sommet 16.

Et enfin celui du 20 peut être parcouru par la branche commençant par le sommet 16 ou celle commençant par le sommet 14.

Et ainsi il existe plusieurs chemins optimales à notre problème et la solution à notre problème n'est pas unique du coup cela revient aux décideurs de choisir.

Exemple:

1ère solution:

s14-s15-s17-s18-s23-s24-s26,

s14-s15-s17-s18-s19-s20-21-s25-s27-s28-s29,

s14-s7-s9,

s14-s7-s8-s13, s12, s16-s30-s31-s34,

s16-s22-s33,

s32,

s10-s11,

s1-s2-s3,

s5-s4-s6

2ème solution:

s16-s30-s31-s34,

s16-s22-s33,

s16-s22-s20-21-s25-s27-s28-s29,

s32,

s14-s15-s17-s18-s23-s24-s26,

s14-s15-s17-s18-s19,

s14-s7-s9,

s14-s7-s8-s13,

s12, s10-s11,

s1-s2-s3,

s5-s4-s6

3 OUTILS UTILISÉE

La résolution du problème a été réalisé avec le langage de programmation Java et la librairie JgraphT. Pour pouvoir exécuter le programme, vous devez ouvrir avec un terminal le dossier TP_GR et ensuite ouvrir le fichier Jar « TP2 » avec la commande :

```
java -jar TP2.jar
```

4 CONCLUSION

Pour conclure, nous avons appris lors de ce TP à :

- Ramener un problème réel à un problème classique de graphe
- Savoir quelle algorithme utilisé pour résoudre un problème de graphe (Etudier la complexité de l'algorithme, savoir dans quelle cas de figure l'algorithme est performant…)
- Faire un arbre couvrant via l'algorithme de Kruskal
- Appliquer nos connaissances sur le recouvrement minimal d'un graphe