الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كليةالتكنولوجيا

قسم: الإعلام الآلي

# Mémoire de Master

## Spécialité : Réseaux Informatiques et Systèmes Répartis

## Thème

# Optimisation des poids dans le routage OSPF par les algorithmes évolutionnaires

**Présenté par :**

BECHAREF Manel

BAHI Amina

**Dirigé par :**

Mr Rahmani Mohamed

Promotion 2021 - 2022

## Abstract

With the growth of the Internet, Internet Service Providers (ISPs) try to meet the increasing traffic demand with new technology and improved utilization of existing resources. Routing of data packets can affect network utilization. Packets are sent along network paths from source to destination following a protocol. Open Shortest Path First (OSPF) is the most commonly used intra-domain Internet routing protocol (IRP). Traffic flow is routed along shortest paths, splitting flow at nodes with several outgoing links on a shortest path to the destination IP address. Link weights are assigned by the network operator. A path length is the sum of the weights of the links in the path. The OSPF weight setting problem seeks a set of weights the optimizes network performance. We study the problem of optimizing OSPF weights, given a set of projected demands, with the objective of minimizing network congestion. The weights assignment problem is an -hard optimization problem. We present a genetic algorithm (GA) to solve this problem. We compare our results with the best known and commonly used heuristics for OSPF weight setting, as well as with a lower bound of the optimal multi-flow flows routing, which is a linear programming relaxation of the problem optimizes weights OSPF routing.

## Résumé

Avec la croissance d'Internet, Internet Service Providers (ISPs) tentent de répondre à la demande croissante de trafic avec de nouvelles technologies et une meilleure utilisation des ressources existantes. Le routage des paquets de données peut affecter l'utilisation du réseau. Les paquets sont envoyés le long des chemins du réseau de la source à la destination en suivant un protocole. Open Shortest Path First (OSPF) est le protocole de routage Internet (IRP) intra-domaine le plus couramment utilisé. Le flux est acheminé le long des chemins les plus courts, en divisant le flux aux nœuds avec plusieurs liens sortants sur un chemin le plus court vers l'adresse IP de destination. Les poids des liens sont attribués par l'opérateur du réseau. La longueur d'un chemin est la somme des poids des liens dans le chemin. Le problème du réglage des poids OSPF cherche un ensemble de poids qui optimise les performances du réseau. Nous étudions le problème de l'optimisation des poids OSPF, étant donné un ensemble de demandes projetées, avec l'objectif de minimiser la congestion du réseau. Le problème de l'affectation des poids est un problème d'optimisation NP-difficile. Nous présentons un algorithme génétique (GA) pour résoudre ce problème. Nous comparons nos résultats avec les heuristiques les plus connues et les plus couramment utilisées pour la détermination des poids OSPF, ainsi qu'avec une limite inférieure du routage optimal des flux multi-flots, qui est une relaxation de programmation linéaire du problème optimise des poids du routage OSPF.

# Remerciements

nous avons eu l'honneur d'être parmi vos élèves et de bénéficier de votre riche enseignement.

Vos qualités pédagogiques et humaines sont pour moi un modèle.

Votre gentillesse, et votre disponibilité permanente ont toujours suscité mon admiration.

Veuillez bien monsieur recevoir mes remerciement pour le grand honneur que vous m  avez fait d'accepter l'encadrement de ce travail.

A Mon Encadreur

**Mr Rahmani Mohamed**

Votre compétence, votre encadrement ont toujours suscité mon profond respect.

Je vous remercie pour votre accueille et vos conseils.

Veuillez trouvez ici, l'expression de mes gratitudes et de ma grande estime.

Je tiens à remercier chaleureusement, tout mes proches et tout ceux qui, de prés ou de loin, m'ont apporté leurs sollicitudes pour accomplir ce Travail.

# Dedicace

Je rends grace à Dieu de m'avoir donner le courage et la volonté.Ainsi que la conscience d'avoir pu terminer mes études.

Je dédie ce modeste travail :

A mes trés chères : A celui qui m'a toujours appris comment réfléchir avant D'agir, à Celui qui m'a soutenu tout long de ma vie scolaire, à Celui qui n'a j'amais épargner un effort

Pour mon bien , Mon cher Père. A celle qui est toujours à coté de mon coeur , à celle qui m'appris le vrai Sens de la vie , à celle qui n'a hésité aucun moment à m'encouragé Ma Chère Mère.

A mes Frères et Ma belle soeur : Imene , Mohammed Amine et le petit AbdelMoudjib pour leur soutien morale. A toute ma famille grande et petite.

A tous mes amis les plus sincères

Amira BOUHAFS et Khadidja ATTAOUI et Rekia CHIKH

A tous les Enseignants et étudiants de Département Mathématique et Informatique .

A tous ceux qui me sont chers, à vous tous

Merci.

***Manel***

# Dedicace

Je dédie cet ouvrage :

A mes trés chères : A celui qui m'a toujours appris comment réfléchir avant D'agir, à Celui qui m'a soutenu tout long de ma vie scolaire, à Celui qui n'a j'amais épargner un effort

A ma chère mère ,
A mon cher père,
Qui n'ont jamais cessé, de formuler des prières à mon égard, de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs

A mes Frères et soeur : Khadidja , Faiza , Moukhtar, Abdelrahime pour leur soutien morale. A toute ma famille grande et petite.

A tous mes amis les plus sincères

Manel BECHAREF et Nacera BENSAKRAN et Aicha DALA.

A tous les Enseignants et étudiants de Département Mathématique et Informatique .

A tous ceux qui me sont chers, à vous tous

Merci.

***Amina***

# Table des matières

# Introduction générale

Les réseaux informatiques ont été développés en réponse à des besoins informatiques, commerciaux et gouvernementaux et pour assurer la compatibilité entre les équipements de différentes sociétés. L'application de normes aux fonctions du réseau a donné lieu à un ensemble de directives afin de créer des matériels et des logiciels pour le réseau, ces logiciels sont composés de protocoles.

Un protocole est une description formelle de règles et de conventions à suivre dans un échange d'informations, que ce soit pour acheminer les données jusqu'au destinataire ou pour que le destinataire comprenne comment il doit utiliser les données qu'il a reçues.

Toutefois, les professionnels des réseaux savent que c'est le routeur qui est responsable du transfert de paquets d'un réseau à l'autre, de la source à la destination.

Un routeur relie plusieurs réseaux. Pour ce faire, il dispose de plusieurs interfaces,chacune possédant une adresse IP différente. Lorsqu'un routeur reçoit un paquet IP sur une interface, il détermine quelle interface utiliser pour transférer le paquet vers sa destination. L'interface utilisée par le routeur pour transférer le paquet peut être le réseau de la destination finale du paquet.

Ce mémoire sera composé en deux parties : une théorique et l'autre pratique. La partie théorique sera décomposé en quatre chapitre suivie par notre implementation en java et nos resultats.En utilisant le dernier chapitre presente nos resultats expérimentaux suivi d'une conclusion générale.

# Chapitre 1

# La Théorie des Graphes

## 1.1 Introduction

Le mot graphique fait référence à une structure mathématique spécifique généralement représentée sous fourme de diagramme constitué de points reliés par des lignes. Dans les applications, les points peuvent, par exemple, correspondre à des atomes chimiques, des villes, des bornes électriques ou tout ce qui peut être connecté par paires. Les lignes peuvent être des liaisons chimiques, des routes, des fils ou d'autres connexions. Les applications de la théorie des graphes sont trouvées dans les communications, les structures et les mécanismes, les réseaux électriques, les systèmes de transport, réseaux sociaux et informatique Les graphes constituent donc une méthode de pensée qui permet de modéliser une grande variété de problèmes en se ramenant à l'étude de sommets et d'arcs. Les derniers travaux en théorie des graphes sont souvent effectués par des informaticiens, du fait de l'importance qu'y revêt l'aspect algorithmique. Nous donnons dans ce chapitre les principales définitions de la théorie des graphes. Celles-ci, bien que formalisées de manière abstraite, sont heureusement très intuitives et donc simples à comprendre. Nous avons également introduit quelques notions sur les algorithmes. En effet, de nombreuses propriétés des graphes peuvent être traduites de manière algorithmique

## 1.2 Graphe et chemins

### 1.2.1 Qu'est-ce qu'un graphe ?

Le plus souvent, les graphes sont un moyen commode pour représenter, modéliser, visualiser certaines situations et problèmes, et pour contribuer à les résoudre. La théorie des graphes est néanmoins une théorie autonome, dans la mesure où elle a ses propres problèmes, et où elle a su développer des outils et concepts, qui lui sont propres, pour les résoudre. Et ces instruments fondamentaux ont de multiples applications dans des domaines très divers. L'exemple le plus classique est la représentation d'un réseau de communication : réseau de routes représenté par une

carte routière, réseau de chemin de fer, de téléphone ou de relais de télévision, réseaux électriques, réseaux d'informations dans une organisation, etc...
Sur le figure 1.1 on represente le graphe G=(S,A)

Exemple : Pendant celui-là vous avez des Sommets et des Arêtes

L'ensemble de Sommet S= $\{A, B, C, D\}$

L'ensemble des Arets $A = \{$(A,D), (D,B), (B,C), (D,C) $\}$

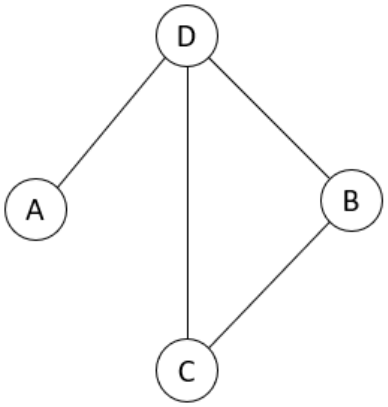
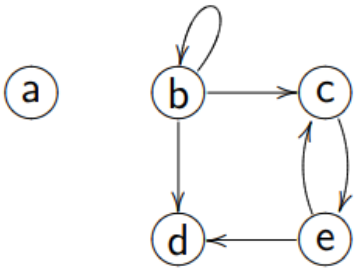



Figure 1.1 :Un graphe

### 1.2.2 Graphes orientés

Un graphe orienté est un couple <S, A>, ou S est un ensemble fini non vide et A une relation sur S. Un élément de S est appelé un sommet ou un nœud et un élément de A est appelé un arc.Sur la figure 1.2 on a un exemple de graphe orienté [1]

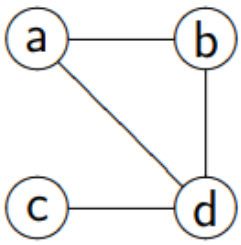



$$G = \langle S, A \rangle$$
où
$$S = \{a, b, c, d, e\}$$
$$A = \{\langle b, b\rangle, \langle b, c\rangle, \langle b, d\rangle, \langle c, e\rangle, \langle e, c\rangle, \langle e, d\rangle\}$$

Figure 1.2 :graphe orienté G

### 1.2.3 Graphes non orientés

Un graphe non orienté est un couple <S, A>, ou S est un ensemble fini non vide et A un ensemble de couples non ordonnés d'éléments de S. Un élément de S est appelé un sommet et un élément de A set appelé une arête. Une arête est représentée par un ensemble de deux sommets, sur le figure 1.3 on a un exemple de graphe non orienté [1]




$$G = \langle S, A \rangle$$
où
$$S = \{a, b, c, d\}$$
$$A = \{\{a, b\}, \{a, d\}, \{b, d\}, \{c, d\}\}$$

Figure 1.3 :graphe non orienté G

### 1.2.4 Terminologies

1) L'ordre d'un graphe |G| est le nombre de ses sommets

2) Une boucle est un arc ou une arête reliant un sommet à lui-même

3) Notions de voisins

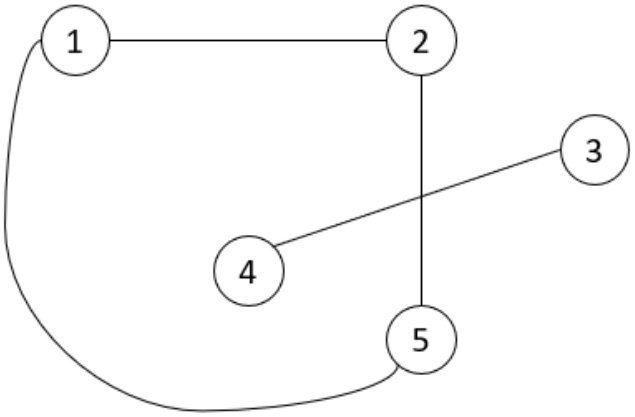
On dit que deux sommets sont VOISINS s'il ont une liaison entre eux


4) Notions de Degré

Le Degré d'un sommet S est le nombre de voisins de S

On note d(S)=N

Par Exemple :



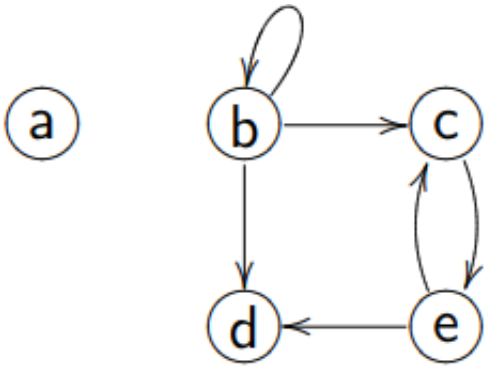

d (1) =2, d (2) =2=d (5) ,d (3) =1, d (4) =1

5) Notions de chemin

Un chemin entre S2 et S2 est constituer par les liaisons et les sommets qui lient S1 et S2

La longueur d'un chemin est le nombre de liaisons le constituant, ⟨c,⟨c,e⟩,e,⟨e,d⟩,d⟩ et

⟨b,⟨b,c⟩,c,⟨c,e⟩,e,⟨e,c⟩,c⟩



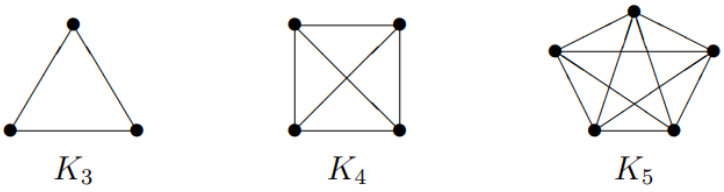

6) Notions de cycle

Un cycle est un chemin fermer c.-à-d. un chemin avec S1=S2

7) Notion de Complétude

Un Graphe complet G (S, L) est un graphe ou tous les sommets sont directement connectés entre eux.

Un graphe complet à S sommets, noté Ks, est un graphe non orienté sans boucle tel qu'il y a une arête entre chaque paire de sommets distincts, par Exemple :



$K_3$ $K_4$ $K_5$

8) Représentation matricielle des graphes

On peut représenter un graphe sur machine avec plusieurs structures de Données : Matrice d'adjacence, Matrice d'incidence, Listes d'adjacence. . .etc.

Chaque représentation présente des avantages et des inconvénients, et leurs utilisations dépendent de problème à résoudre

8.1) La matrice d'adjacence pour un graphe non orienté

A tout graphe d'ordre |X| = n, on associe une matrice M de n lignes et n colonnes dont les éléments sont notés Mij :

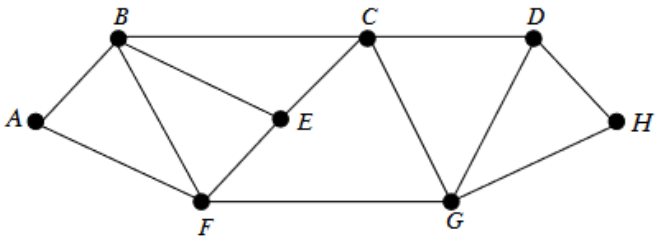
Pour un graphe non orienté G= (X, E)

$$\mathrm{M}_{ij}= \begin{cases} 1 & \text{si } \{\mathrm{x}_i = x_j\ \} \in \mathrm{E} \\ 0 & \text{sinon} \end{cases} \tag{1.1}$$

Degré d'un sommet S :

$$d(s) = \sum M_S j$$

On se donne ce graphe G suivant




| A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

8.2) La matrice d'adjacence pour un graphe orienté

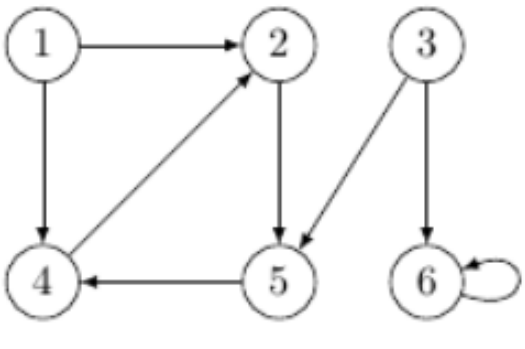
A tout graphe d'ordre |X| = n, on associe une matrice M de n lignes et n colonnes dont les éléments sont notés $M_ij$ :

Pour un graphe non orienté $G = (X, U)$

$$\mathrm{M}_{ij}= \begin{cases} 1 & \text{si } \{\mathrm{x}_i = x_j \} \in \mathrm{U} \\ 0 & \text{sinon} \end{cases} \tag{1.2}$$

Degré d'un sommet $S : d(s) = d^+(s) + d^-(s)$

On se donne le graphe G suivant



| | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |

Figure 1.4 : Représentation en Matrice D'adjacence

## 1.3 Problèmes de plus court chemin

La recherche du meilleur itinéraire que ce soit en distance, en temps ou en coût d'un point à un autre peut être modélisée par la recherche du plus court chemin dans un graphe [1].

Dans ce paragraphe, on s'intéresse à la recherche d'un plus court chemin dans un graphe entre deux sommets donnés

### 1.3.1 Graphe pondéré

On appelle graphe pondéré, un graphe (orienté ou non) dont les arêtes ont été affectées d'un nombre appelé poids (ou coût) [2]

On peut définir la matrice des poids $P(a_ij)$ du graphe , dont les coefficients $a_ij$ correspondent aux poids des arête

$$\mathrm{a}_{ij}= \begin{cases} 0 \text{ si i = j} \\ \infty \text{ s'il n'existe pas d'arete(ou d'arc)entre les sommets } x_i \text{ et} x_j \\ \mathrm{P}_(i,j) \text{ ou } P_(i,j) \text{ est le poids de l'arete( ou de l'arc)entre les sommets } x_i \text{ et } x_j \end{cases} \tag{1.3}$$

On se donne ce graphe G de figure 1.5

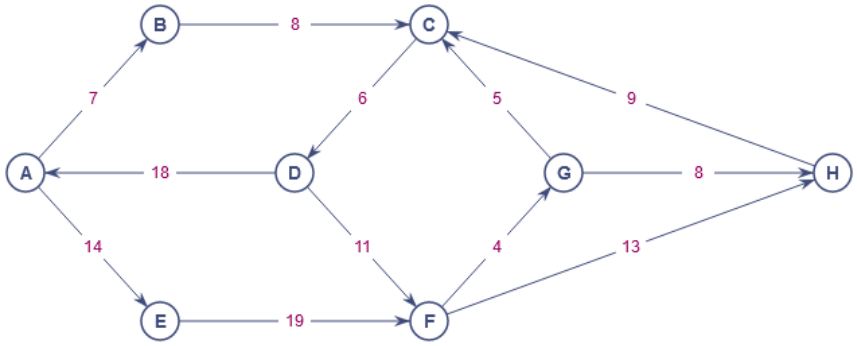



Figure 1.5 : Un graphe orienté pondéré

Les sommets du graphe étant rangés dans l'ordre alphabétique la matrice des poids est

$$P = \begin{pmatrix} 0 & 7 & \infty & \infty & 14 & \infty & \infty & \infty \\ \infty & 0 & 8 & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & 0 & 6 & \infty & \infty & \infty & \infty \\ 18 & \infty & \infty & 0 & \infty & 11 & \infty & \infty \\ \infty & \infty & \infty & \infty & 0 & 19 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 4 & 13 \\ \infty & \infty & 5 & \infty & \infty & \infty & 0 & 9 \\ \infty & \infty & 9 & \infty & \infty & \infty & \infty & 0 \end{pmatrix}$$

### 1.3.2 Algorithme de Dijkstra

#### Description de l'Algorithme de Dijkstra

L'algorithme de Dijkstra, développer par E. W. Dijkstra en 1959. Permet de calcul du plus court chemin dans un graphe .

Si on souhaite déterminer le plus court chemin entre deux sommets d'un graphe, on peut essayer d'énumérer tous les chemins possibles entre ces deux sommets et calculer leurs longueurs.

**Algorithm 1** Dijkstra (G,S)

Entrées :G = (S, A, W) un graphe pondéré, Sun sommet de G

Sortie : Un dictionnaire qui à chaque sommet associe sa distance à S

1 $F := S, \{initialement Fcontient tous les sommets\}$

2 Pour chaque sommet u S Faire

3 $d[u] := +\infty$

4 $d[s] := 0$

5 Tant que $F \neq \emptyset$ Faire

6 $u := Extraire_L E_M IN(F)$

7 Pour chaque sommet $v \epsilon V[u]$Faire

8 $Sid[v] > d[u] + W(u, v)$Alors

9 $d[v] = d[u] + W(u, v)$

10 Fin Pour

11 Fin Tant que

12 Renvoyer d

Exemple :

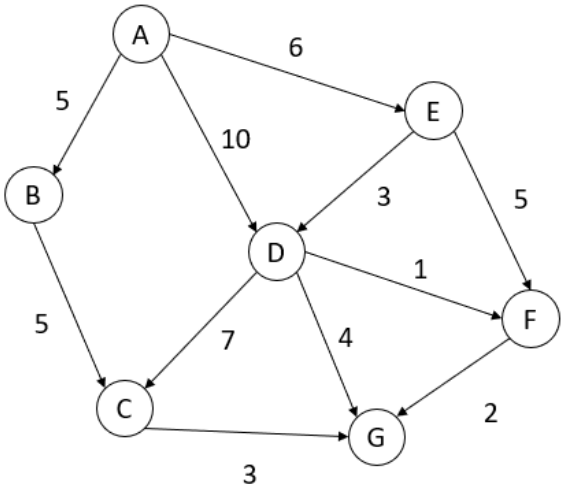



Figure 1.6 : Recherche les plus courts chemins issus du sommet

Donc la parcours qui permet trouver le plus court chemin à partir de sommet A vers sommet G En rencontrant le minimum d'obstacles est A-E-D-F-G

TABLE 1.1 – Tableau résumant les étapes de l'algorithme de Dijkstra

| A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|
| 0 | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ |
| X | 5,A | $\infty$ | 10,A | 6,A | $\infty$ | $\infty$ |
| X | X | 10,B | 10,A | 6,A | $\infty$ | $\infty$ |
| X | X | 10,B | 9,E | X | 11,E | $\infty$ |
| X | X | 10,B | X | X | 10,D | 13,D |
| X | X | X | X | X | 10,D | 13,D ou 13,C |
| X | X | X | X | X | X | 12,F |

## 1.4 Introductions et Notions de base des flots

### 1.4.1 Introduction

Un graphe orienté qui a un certain nombre de propriétés de caractéristique. Tout d'abord il est composé d'un certain nombre de sommet interne puisse de sommet, particuliers l'on est la source S et l'autre qui est puits T.

On va aussi faire d'autre Hypothèses simplificatrices pour éviter d'écrire de grosse formule. Hypothèses pour simplifier :

S n'a pas d'arcs entrants : donc il n'y a que des arcs sortants qui vont de S vers d'autres sommets

T n'a pas d'arcs sortant : c.-à-d. que tu es n'a que des arcs entrants Maintenant sur chacun des arcs qu'on va mettre une étiquette par exemple une valeur qui si votre entière mais ce n'est pas une obligation

cette étiquette c'est ce que l'on appelle la capacité d'un arc

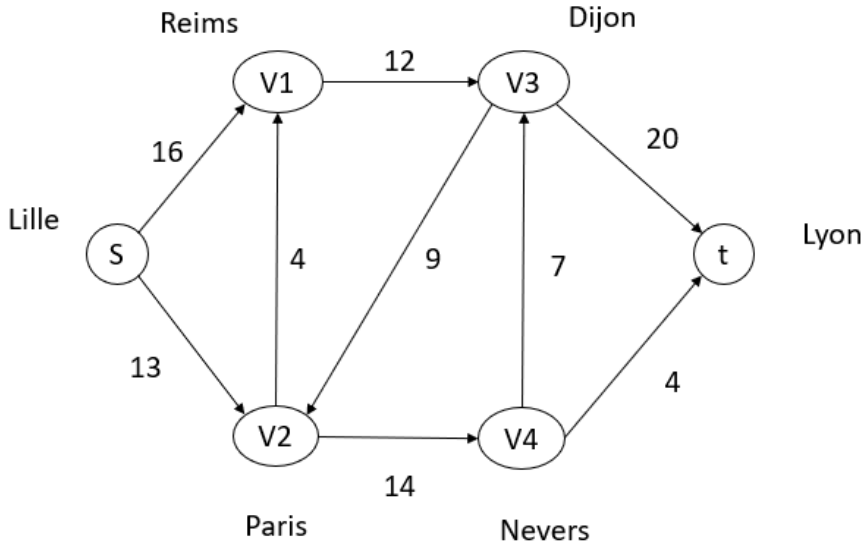



Figure :Exemple Flots et réseaux de transport

### 1.4.2 Réseaux de flot

Un graphe orienté G, avec deux sommets spéciaux

L'origine (une source) S

La destination (un puits) T

Et pour chaque arc une capacité c : E(G)→$R^+$

### 1.4.3 Flots

Un Flot G sur un réseaux de flot est une Fonction $E(G)ßR^+$ qui satisfait aux deux propriétés suivantes :

Contrainte de capacité : Quantité de flot sur arc capacité de l'arc

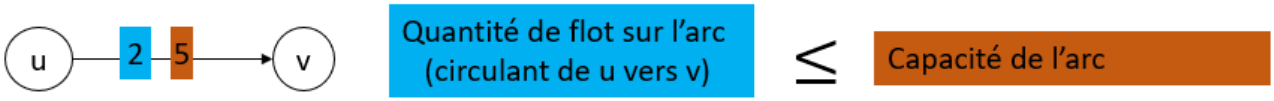



Figure 1.9

Conservation du flot : pour chaque sommet interne du réseau (sauf à la source et au puits) le flot entrant est égal au flot sortant

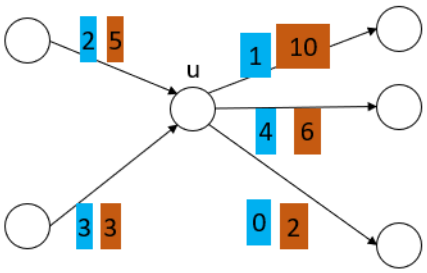



Figure 1.10

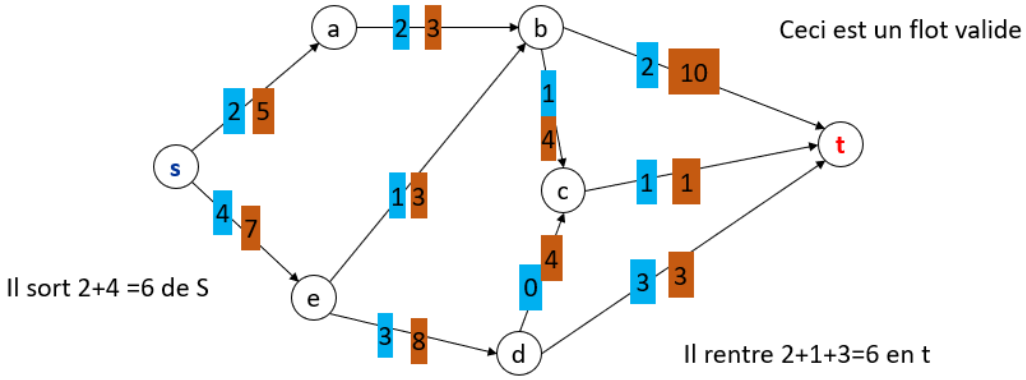



Figure 1.11

La valeur d'un flot est définie comme quantité de flot qui entre en T (ou quantité de flot qui sort de S)

### 1.4.4 Algorithme de Ford-Fulkerson

#### Objectif

La Méthode de Ford-Fulkerson qui permet de résoudre le problème du flot maximum

**Algorithm 2** Ford-Fulkerson (G,S,T)

1 Pour chaque arc $(u.v)[G]$
2 Faire $f[u.v] \leftarrow 0$
3 $f[v,u] \leftarrow 0$
4 Tant qu'il existe un chemin p de S à T dans le réseau résidel $G_f$
5 Faire $c_f(p) \longleftarrow min\{\ c_f(u,v) : (u,v)$ is in p $\}$
6 Pour chaque arc (u,v) de p
7 Faire f[u,v] $\longleftarrow [u,v] + c_f(p)$
8f[v,u] $\longleftarrow [u,v]$

### 1.4.5 Algorithme de Dijkstra par Tas binaire

#### File de Priorités

Définition :

Une file est une structure de données avec les opérations enfiler (ajouter un élément) et défiler (enlever le plus vieil élément).

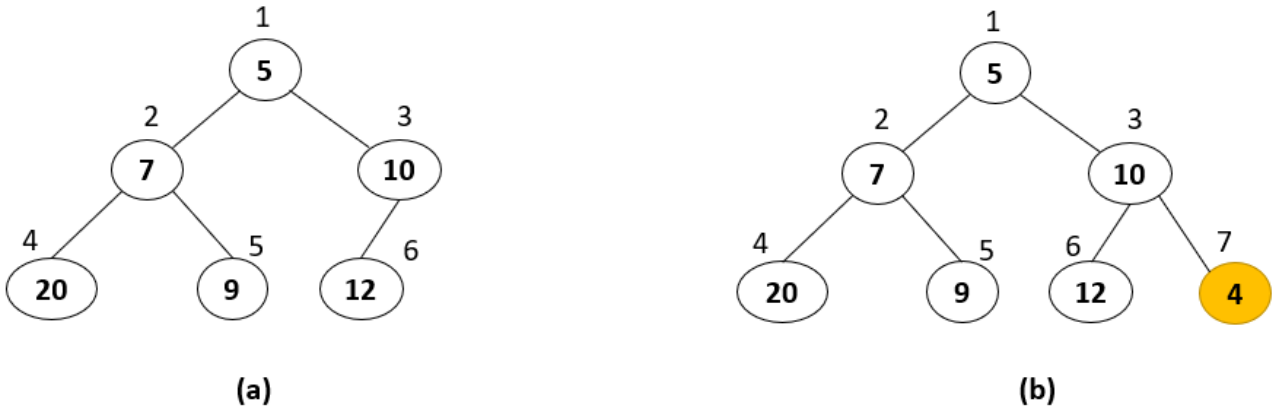
L'objectif des files de priorité est stocké des éléments munis d'une clé et pouvoir :

Tester si la file de priorité est vide

Ajouter un élément (et sa clé)

Extraire un élément ayant une clé min

Exemple Ajouté une clé figure 1.12 :



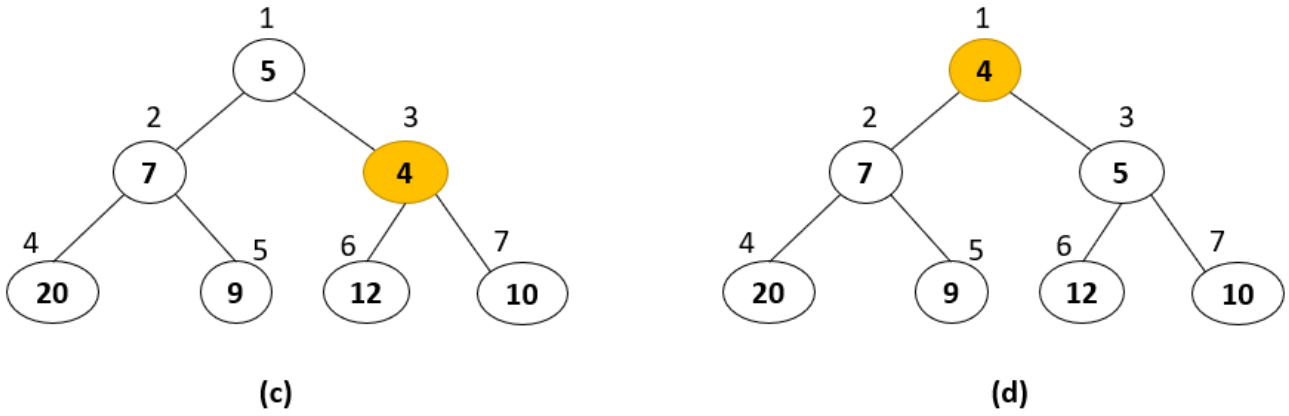



Figure 1.12 : La circulation des valeurs pendants l'Addition de 4

**Définition d'un Arbre**

Un arbre est schématisé (modélisé) par des nœuds dont chacun possède éventuellement des descendants (fils),Chaque nœud possède au plus un père,Le nœud ne possédant pas de père s'appelle racine de l'arbre. (En haut de l'arbre,Un nœud ne procédant pas de fils est une feuille

### 1.4.6 Arbre Binaire

Définition :
Arbre ou chaque nœud possède aux plus deux fils. On parle de fils gauche (FG) et fils droit (FD)

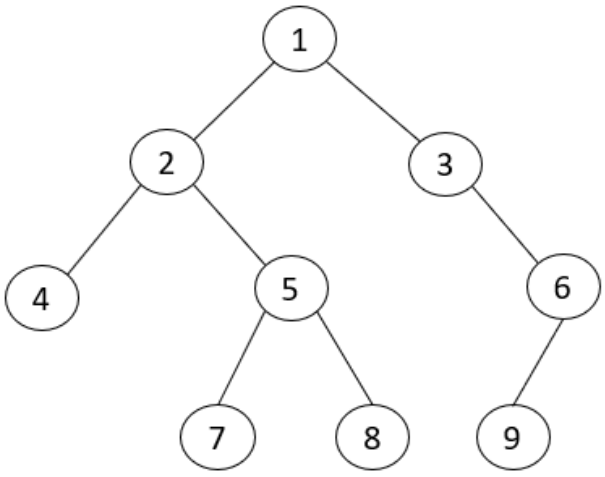



Figure 1.13 :Arbre binaire

La Profondeur d'un nœud p dans un arbre est la longueur (nombre d'arrêts) du chemin à partir de la racine jusqu'au nœud p

La hauteur d'un arbre binaire est la profondeur maximale d'un nœud, ou -1 si l'arbre est vide

### 1.4.7 Arbre binaire de recherche

Définition :
Les arbres binaires de recherche sont utilisés pour accélérer la recherche dans les arbres. Un arbre binaire de recherche est un arbre binaire vérifiant la propriété suivante.

Soient x, y deux nœuds de l'arbre, si y est u nœud de sous-arbre gauche de x, alors clé(y) clé (x), si y est un nœud du sous-arbre droit de x alors clé(y)  clé(x)

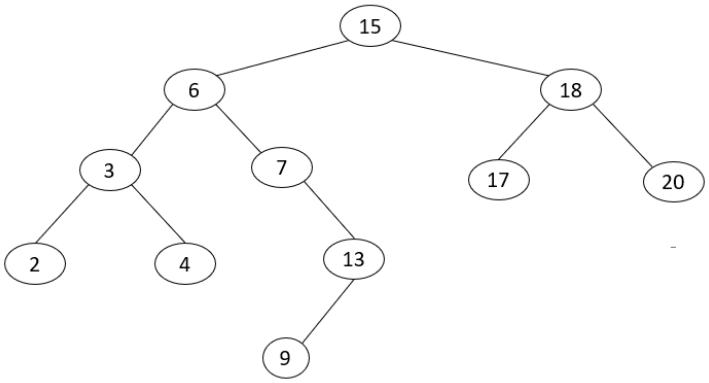



Figure 1.14 :Arbre binaire de recherche

### 1.4.8 Arbre Binaire Parfait

Définition :

Un arbre binaire est parfait si :

Tous les nœuds de chaque niveau sont présents sauf éventuellement au dernier niveau

Les feuilles du dernier niveau doivent être regroupées de la gauche de l'arbre

### 1.4.9 Structure de Tas ou (Heap)

Un tas binaire est schématisé par un arbre binaire presque-complet, ou chaque nœud est prioritaire par rapport à ses fils

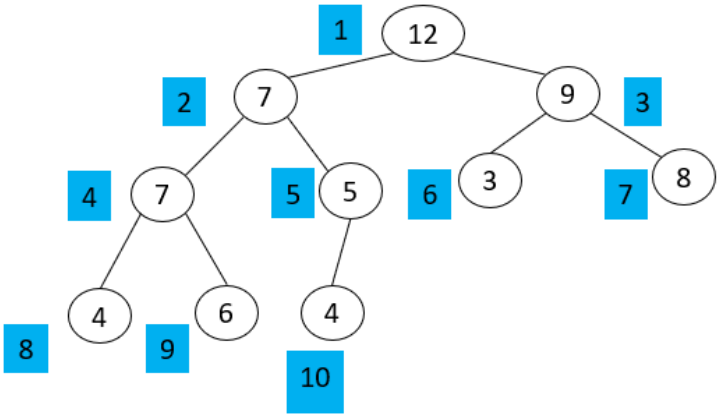
S'il ya n nœuds, la hauteur est [ $\log_2 n$]

Il est Implémenté par un tableau T tel que pour chaque élément T[i] :

- Le fils gauche, s'il existe, est à la position 2*i+1

- Le fils droit, s'il existe, est à la position 2*i+2

- Le père, exceptée la racine, est à la position (i-1) /2




Représentation d'un arbre binaire à l'aide d'un tas

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| 12 | 7 | 9 | 7 | 5 | 3 | 8 | 4 | 6 | 4 |

### 1.4.10 Algorithme de Dijkstra par Tas binaire

On presente ci dessus l'algorithme de dijkstra par tas binaire qu'on implemente dans notre travail

**Algorithm 3** Dijkstra par Tas Binaire

Initialisation :

1 On Pose $l(s) = 0$; marque(s)=vrai et ;
2 Pour tout $x \neq s$ ; $l(x)$=$+\infty$;
3 Pred $(x) = -1$ et marque (x)=faux ;
4 On prend $H$= $\{s\}$;

Itération :
5 Tant que $H \neq \emptyset$
6 Faire
7 $x$ :=FindDeleteMin($H$) ; marque[$x$] =vrai
8 Pour chaque sommet $y \epsilon$ Succ($x$)
9 Faire
10 Si not (marque [$y$]) et $l[y] > l[x] + d(x, y)$ Alors
11 $l[y] := l[x] + d(x, y)$ ;
12 $pred[y] := x$ ;
13 AddHeap($H, y$) ;

## 1.5 Conclusion

Ce chapitre a présenté les différentes notions de théorie de graphes, qu'est-ce un graphe et définir un petit peu de vocabulaire qui sera utile pour comprendre les théories de graphes .et enfin nous avons achevé ce chapitre par de fonctionnement de Algorithmes de Dijkstra.

# Chapitre 2

# Généralités sur les réseaux

## 2.1 Introduction

Dans ce chapitre nous allons tout d'abord étudier les définitions théoriques nécessaires des notions fondamentales d'un réseau informatique ainsi que le but des réseaux informatique et leurs avantages et inconvénients.

Un réseau est un groupe d'ordinateurs connectés les uns aux autres dans le but de partager des informations ou des ressources, qui respecte un ensemble de protocoles

Mais si l'on veut connecter un groupe d'ordinateurs fabriqués par différents constructeurs, il faut représenter l'information en général, compréhensible et reconnue par tous (les constructeurs) dans le but de connecter deux ordinateurs de différents constructeurs Voici le concept OSI Model et TCP IP Modèle [7]

## 2.2 Définition d'un réseau informatique

Ensemble d'ordinateurs et de terminaux interconnectés pour échanger des informations numériques Un réseau est un ensemble d'objets interconnectés les uns avec les autres. Il permet de faire circuler des éléments entre chacun de ces objets selon des règles bien définies

Il existe deux types de réseaux : les réseaux filaires et les réseaux sans fil[5] :

• Le réseau filaire est un réseau que l'on utilise grâce à une connexion avec fil. Ce réseau utilise des câbles pour relier des périphériques et des ordinateurs.

• Le réseau sans fil est un réseau qui n'utilisent pas de câbles. C'est une technique qui permet aux particuliers, aux réseaux de télécommunication et aux entreprises de limiter

l'utilisation de câbles

## 2.3 Les objectifs d'un réseau informatique

Un réseau informatique permet [3]

— Partages des ressources matérielles et logicielles

— La partage de fichiers, d'applications et de ressources

— La Communication entre personne (courrier électronique, discussion en direct...)

— a communication entre processus (entre des machines industrielles)

— Recherche d'information : Internet

## 2.4 Les différents types de réseaux

On peut distinguer différente catégorie des réseaux informatiques selon plusieurs critère tel que [3] :

— La taille de réseau

— itesse de transfert de données

— Leur étendu

On fait trois types de réseaux selon leur étendue :

- LAN (Local Area Network)
- MAN (Metropolitain Area Network)
- WAN (Wide Area Network)

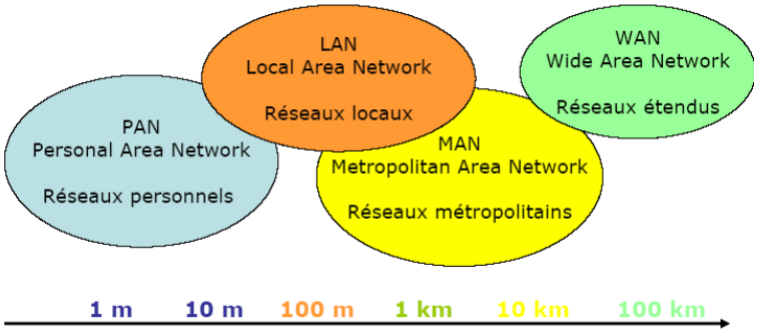



Figure 2.1 :Classification des réseaux selon leur taille

### 2.4.1 Les réseaux locaux (LAN)

Il s'agit d'un ensemble d'équipements (ordinateurs, imprimantes...) appartenant à une même organisation et relies entre eux dans une petite aire géographique par un réseau, il ne dépasse pas généralement les centaines de machines et un kilomètre de distance, la vitesse de transmission vont de 10 à 100 MB/S (Mégabits par seconde) [3]

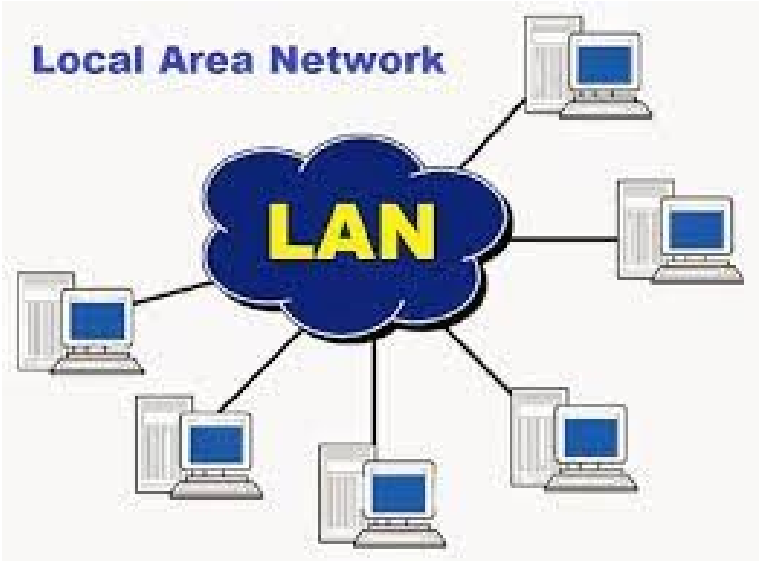



Figure 2.2 : Réseau local (LAN)

### 2.4.2 Les réseaux métropolitains (MAN)

Il sert à interconnecter des réseaux LAN distants de quelques kilomètres. Ainsi un MAN permet à deux nœuds distants de communiquer comme si lis faisaient partie d'un même réseau local. Un MAN est formé de commutateurs ou de routeurs interconnectés par des liens hauts débits (en général en fibre optique) [3]

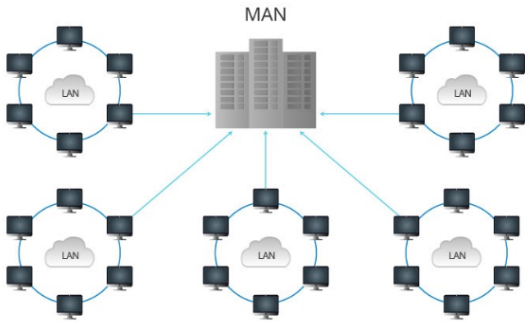



Figure 2.3 : Réseau Métropolitain (MAN)

### 2.4.3 Les réseaux étendus (WAN)

Un WAN interconnecte plusieurs LANs à travers de grandes distances géographiques. Il couvre une grande zone géographique, typiquement à l'échelle d'un pays d'un continent, ou de la planète entière [3].

Les débits disponibles sur un WAN résultent d'un arbitrage avec le cout des liaisons.

Les WAN fonctionnent grâce à des routeurs qui permettent de choisir le trajet le plus approprié pour atteindre un nœud du réseau, le plus connu des WAN et l'internet

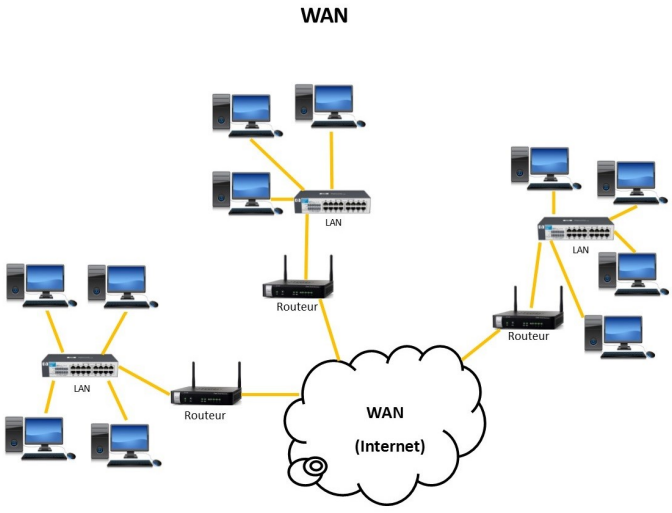



Figure 2.4 : Réseau étendus (WAN)

## 2.5 Topologie physique des réseaux

La topologie décrit la manière dont les équipements réseaux sont connectés entre eux Il y a trois types de topologie

### 2.5.1 Topologie en bus

Un réseau en bus est une architecture de communication ou la connexion des matériels est assurée par un bus partagé par tous les utilisateurs. Lorsqu'une station émet des données, elles circulent sur toute la longueur du bus et la station destinatrice peut les récupérer. Une seule station peut émettre à la fois. En, bout de bus, un « bouchon » permet de supprimer définitivement les informations pour qu'une autre station puisse émettre [6].

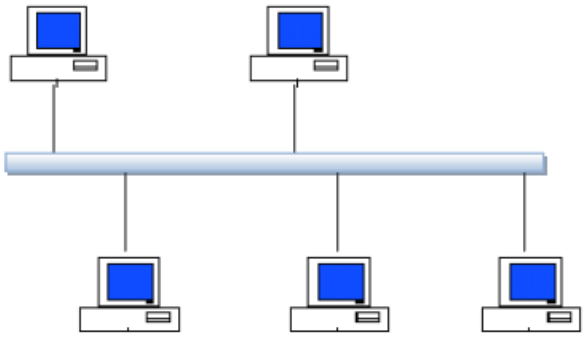


Figure 2.5 :Topologie en Bus

### 2.5.2 Topologie en étoile

C'est la topologie la plus courante actuellement. Dans une topologie en étoile, tous les câbles sont raccordés à un point central, par exemple un concentrateur ou un commutateur. Elle est aussi très souple en matière de gestion et dépannage de réseau :la panne d'un poste ne perturbe pas le fonctionnement global du réseau [6].

L'inconvénient principal de cette topologie réside dans la longueur des câbles utilisés

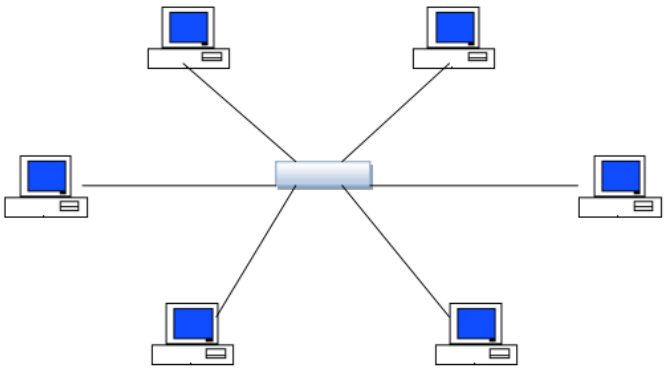


Figure 2.6 :Topologie en étoile

### 2.5.3 Topologie en anneau

Dans une topologie en anneau, chaque hôte est connecté à son voisin. Le dernier hôte se connecte au premier. Cette topologie crée un anneau physique de câble,elle utilise la méthode d'accès à "jeton" (Token ring). Les données transitent de stations en stations en suivant l'anneau qui chaque fois régénère le signal. Le jeton détermine quelle station peut émettre, il est transféré à tour de rôle vers la station suivante [6].

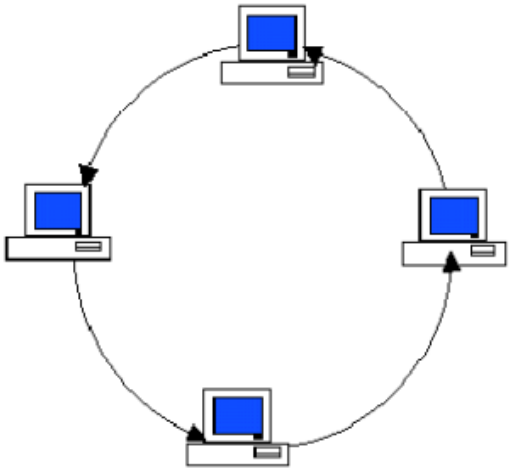


Figure 2.7 :Topologie en anneau

### 2.5.4 Topologie Arborescente

Une topologie en arbre ou topologie arborescente ou hiérarchique c'est une structure arborescente hiérarchisée peut être considérée comme une collection de réseaux en étoile reliés entre eux par des concentrateurs jusqu'à un nœud unique central. Cette structure est très utilisée dans les réseaux locaux [6]

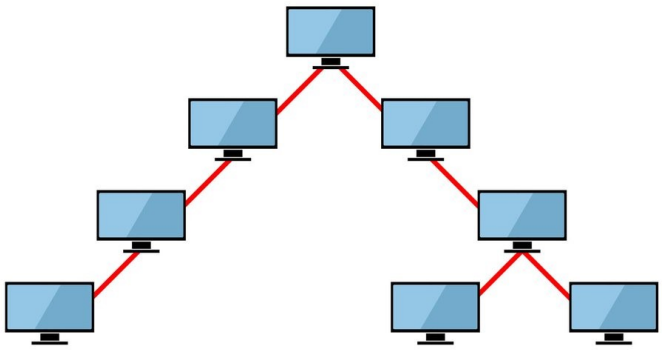


Figure 2.8 :Topologie en arbre

## 2.6 Les Architectures de réseaux

### 2.6.1 Définition

On appelle l'architecture de communication tous les éléments logiciels et matériels à utiliser pour que deux machines communiquent entre eu.

L'architecture de communication est organisée selon une hiérarchie des couches. Chaque couche représente un service, cette architecture de couches respecte deux règles responsables :

- Chaque couche demande un service de la couche inférieur
- Chaque couche ne peut communiquer qu'avec la couche du même niveau dans un autre système

### 2.6.2 Modèle OSI

En 1984, L'Organisation Internationale de la Standardisation 'ISO' a publié un modèle de référence pour l'Interconnexion du Systèmes Ouverts 'OSI' (Open System Interconnexion)

Le modèle OSI présente la structure en couche, chaque couche propose et fournie des services à la couche qui lui est immédiatement supérieur, le modèle OSI est composé de 7 couches qui regroupement l'ensemble des services nécessaires

Les couches du modèle OSI sont :

- Couche Physique
- Couche Liaison de données
- Couche Réseau
- Couche Transport
- Couche Session
- Couche Présentation
- Couche Application

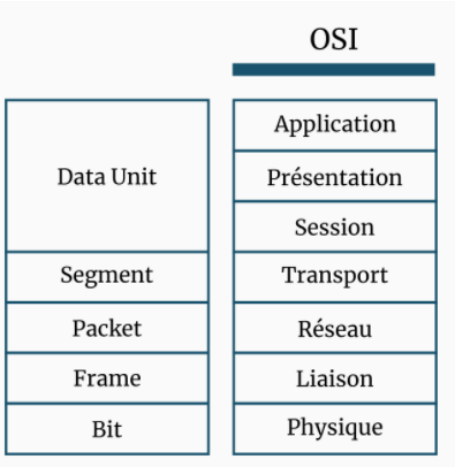



Figure 2.9 :Modèle OSI

### La Couche physique

La couche physique vous aide à définir les spécifications électriques et physiques de la connexion de données. Ce niveau établit la relation entre un appareil et un support de transmission physique. La couche physique n'est pas concernée par les protocoles ou d'autres éléments de couche supérieure
Des exemples de matériel dans la couche physique sont les adaptateurs réseau, Ethernet, les répéteurs, les concentrateurs de réseau . . .etc.

### La Couche de liaison de données

La couche liaison de données établit et termine une connexion entre deux nœuds physiquement connectés sur un réseau. Il divise les paquets en trames et les envoie de la source à la destination. Cette couche est composée de deux parties : le contrôle de liaison logique (LLC), qui identifie les protocoles réseau, effectue la vérification des erreurs et synchronise les trames, et le contrôle d'accès au support (MAC) qui utilise les adresses MAC pour connecter les appareils et définir les autorisations de transmission et de réception de données [6].

### La couche Réseau

La couche réseau a deux fonctions principales. L'une consiste à diviser les segments en paquets réseau et à réassembler les paquets à l'extrémité de réception. L'autre consiste à acheminer les paquets en découvrant le meilleur chemin à travers un réseau physique. La couche réseau utilise des adresses réseau (généralement des adresses de protocole Internet) pour acheminer les paquets vers un nœud de destination [6].

### La couche transport

La couche de transport prend les données transférées dans la couche de session et les divise en « segments » à l'extrémité de transmission, Il est chargé de réassembler les segments à l'extrémité réceptrice, les retransformant en données pouvant être utilisées par la couche de session. La couche de transport effectue un contrôle de flux, en envoyant des données à un débit qui correspond à la vitesse de connexion du périphérique de réception, et un contrôle d'erreur, en vérifiant si les données ont été reçues de manière incorrecte et si ce n'est pas le cas, en les redemandant

### La couche session

La couche session crée des canaux de communication, appelés sessions, entre les appareils. Il est chargé d'ouvrir les sessions, de s'assurer quelles restent ouvertes et fonctionnelles pendant le transfert des données et de les fermer à la fin de la communication. La couche session peut également définir des points de contrôle lors d'un transfert de données, si la session est interrompue, les appareils peuvent reprendre le transfert de données à partir du dernier point de contrôle.

### La couche présentation

La couche présentation prépare les données pour la couche application. Il définit comment deux appareils doivent coder et compresser les données afin qu'elles soient reçues correctement à l'autre extrémité. La couche de présentation prend toutes les données transmises par la couche d'application et les prépare pour la transmission sur la couche de session.

### La couche Application

La couche d'application est utilisée par les logiciels de l'utilisateur final tels que les navigateurs Web et les clients de messagerie. Il fournit des protocoles qui permettent aux logiciels d'envoyer et de recevoir des informations et de présenter des données significatives aux utilisateurs. Quelques exemples de protocoles de couche application sont HyperText Transfer Protocol (HTTP), File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP) et Domain Name System (DNS).

## 2.7 Le Modèle TCP /IP

TCP/IP signifie Transmission Control Protocol/Internet Protocol et est une suite de protocoles de communications utilisés pour interconnecter des périphériques réseau sur Internet. TCP/IP est également utilisé comme protocole de communication dans un réseau informatique privé.

L'ensemble de la suite IP un ensemble de règles et de procédures est communément appelé TCP/IP. TCP et IP sont les deux principaux protocoles, bien que d'autres soient inclus dans la suite. La suite de protocoles TCP/IP fonctionne comme une couche d'abstraction entre les applications Internet et la structure de routage et de commutation.

La fonctionnalité TCP/IP est divisée en 4 couches, chacune comprenant des protocoles spécifiques :

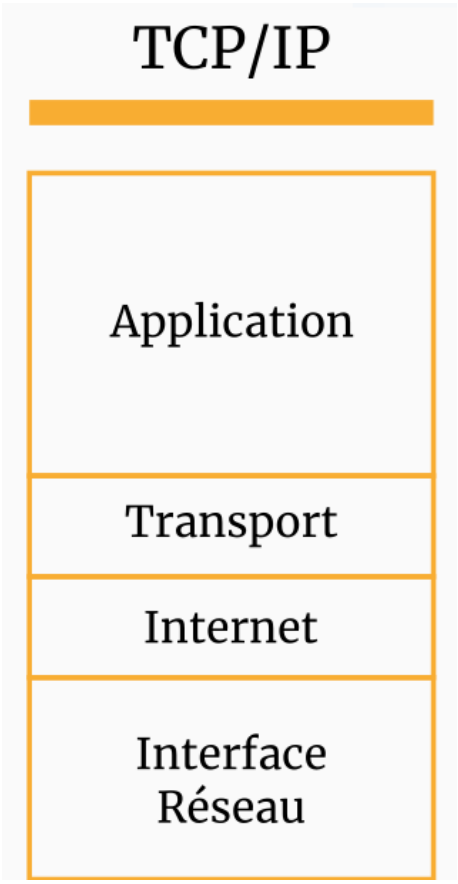



Figure 2.10 :Modèle TCP/IP

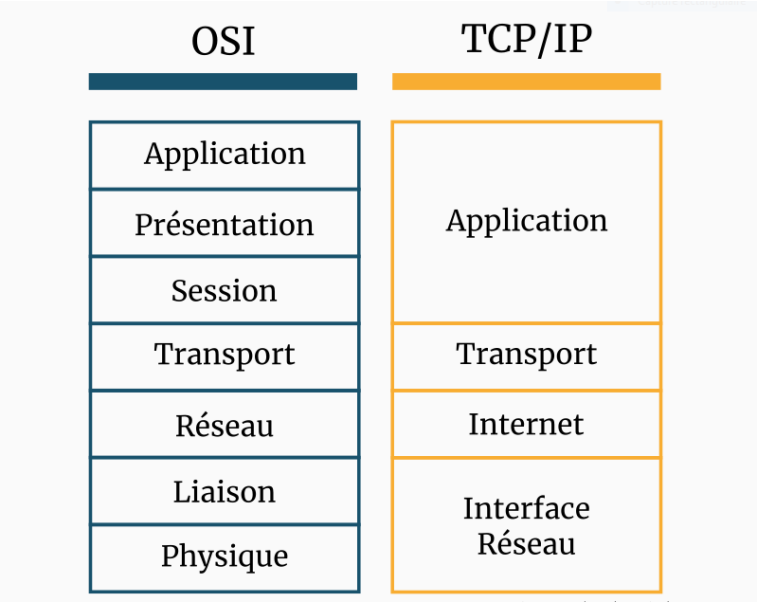



Figure 2.11 :Le Modèle TCP/IP et le modèle OSI

Comme on peut le remarquer, les couches du modèle TCP/IP ont des taches beaucoup plus diverses que les couches du modèle OSI, étant donné que certaines couches du modèle TCP/IP correspondent à plusieurs couches du modèle OSI

Les rôles des différentes couches du modèle sont les suivants

### 2.7.1 La couche Application

La couche application fournit aux applications un échange de données standardisé. Ses protocoles incluent HTTP, FTP, Post Office Protocol 3, Simple Mail Transfer Protocol et Simple Network Management Protocol. Au niveau de la couche d'application, la charge utile correspond aux données d'applications réelles.

### 2.7.2 La couche Transport

La couche transport est responsable du maintien des communications de bout à travers le réseau. TCP gère les communications entre les hôtes et fournit le contrôle de flux, le multiplexage et la fiabilité. Les protocoles de transport incluent TCP et User Datagram Protocol, qui est parfois utilisé à la place de TCP à des fins spéciales

### 2.7.3 La couche Internet

La couche réseau, également appelée couche Internet, traite les paquets et connecte des réseaux indépendants pour transporter les paquets à travers les limites du réseau. Les protocoles de couche réseau sont IP et Internet Control Message Protocol, qui est utilisé pour le rapport d'erreurs.

### 2.7.4 La couche Accès réseau (Hôte réseau)

Également connue sous le nom de couche d'interface réseau ou couche de liaison de données, se compose de protocoles qui fonctionnent uniquement sur une liaison -le composant réseau qui interconnecte les nœuds ou les hôtes du réseau. Les protocoles de cette couche incluent Ethernet pour les réseaux locaux et Address Resolution Protocol

## 2.8 Les équipements réseau

Les équipements d'interconnexion sont les matériels qui nous permettent de relier les équipements. Il y a plusieurs types d'équipements en réseau informatique :

Routeur, commutateur (Switch), concentrateur (Hub), FireWall, Répéteur, Multiplexeur, Voice devices (IP Phone ...), Wireless Access Point (WAP), Load Balancer, etc.....

### 2.8.1 Répéteur

Un répéteur est un équipement simple permettant de régénérer un signal entre deux nœuds du réseau, afin d'étendre la distance du câblage

C'est un équipement simple qui opère au niveau 1 du modèle OSI (couche physique), et qui ne nécessite aucune administration

### 2.8.2 Le concentrateur (HUB)

Le concentrateur est un périphérique qui opère niveau 1du modèle OSI (couche physique), est un appareil qui répète les signaux qu'il reçoit sur un port vers tous les autres ports. C'est une centrale point de connexion pour plusieurs périphériques réseau

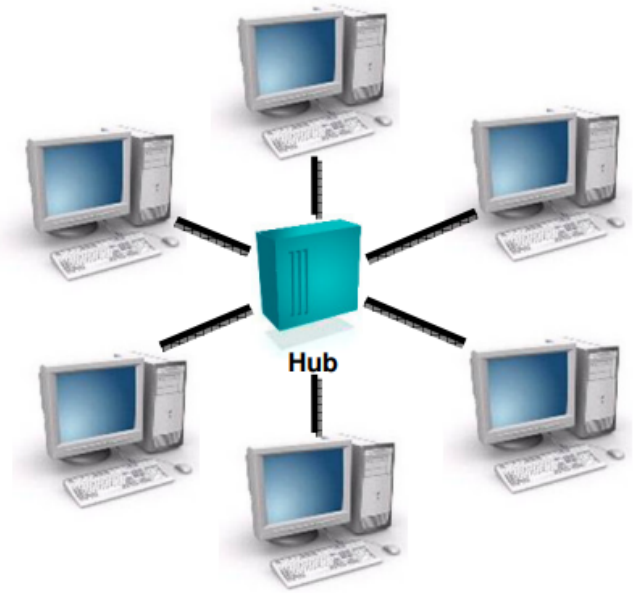



Figure 2.12 :Le concentrateur

### 2.8.3 Le commutateur (Switch)

Le commutateur capable d'analyse les trames arrivant sur ses ports d'entrée et filtre les données. Ils sont aussi appelés Switch et travaillent au niveau 2 du modèle OSI.

Les commutateurs introduits pour augmenter la bande passante globale d'un réseau. Il possède une table MAC contenant les n° de ports et Adresses Mac, chaque port du commutateur est un domaine de collisions

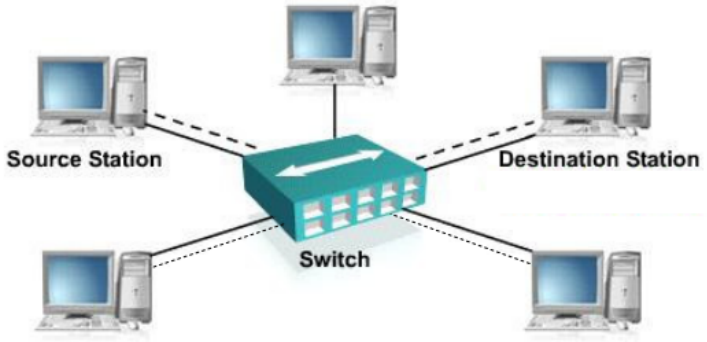



Figure 2.13 :Le commutateur

### 2.8.4 Le Routeur

Un routeur est un équipement d'interconnexion de réseaux informatiques permettant d'assurer le routage des paquets entre deux réseaux, en choisissant le chemin selon un ensemble

de règles formats la table de routage.

Le routeur possède une table de routage contenant les numéros des ports et les adresses IP, il opère au niveau 3 du modèle OSI (couche réseau)

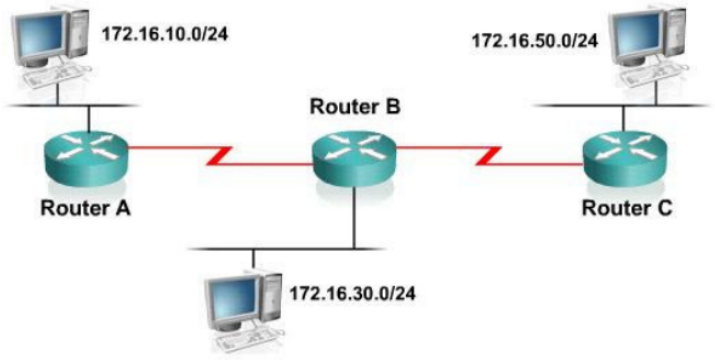



Figure 2.14 :Le Routeur

### 2.8.5 Firewall

La plupart des firewalls travaillent au niveau de couche 4, il a pour but de protège un réseau des accès non -autorisés, il contrôle le trafics entrant et sortant du réseau sur la base de certains paramètres

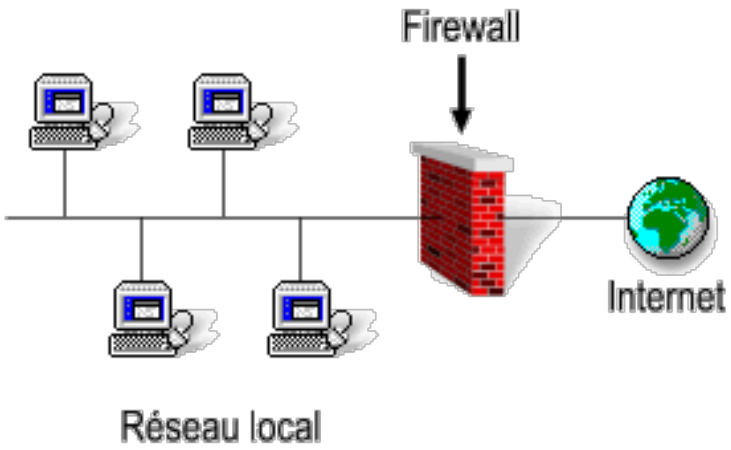



Figure 2.15 :Firewall

### 2.8.6 Access Point

Dans les réseaux informatiques Access point qui permet un point de consolidation pour les clients, relie les domaines sans fil et câblés

Access point en tant que dispositif autonome est habituellement relié à un routeur, mais il peut aussi faire partie intégrante du routeur lui même

Access point différenciateurs

— Géré de manière centralisée (LightWeight)

— Géré individuellement (Aotonomous or standalone)

— Quantités et type d'émetteurs-récepteurs (2,4 GHz, 5GHz)

— Fonctionnalités améliorées et exclusives

## 2.9 Les Protocoles

Un protocole réseau est un ensemble de règles et de procédures de communication qui régissent les échanges de données entre les appareils, décrit de manière formelle

Chaque protocole réseau a sa propre fonction, son propre format et ses propres règles de communication. Il existe de nombreux protocoles réseaux

Voici quelques protocoles les plus utilisés :

### 2.9.1 TCP (Transfer Control Protocol)

TCP est un protocole de communication utilisé pour communiquer sur un réseau. Il divise tout message en séries de paquets qui sont envoyés de la source à la destination. TCP est un protocole de couche transport fournit une vérification approfondie des erreurs

### 2.9.2 IP (Protocole Internet)

IP est un protocole de couche réseau qui contient des informations d'adressage et de contrôle. Il est principalement utilisé avec TCP

TCP/IP est le protocole le plus répandu pour connecter les réseaux

### 2.9.3 UDP (User Datagram Protocol)

UDP est un protocole de communication de substitution au protocole de contrôle de transmission mis en œuvre principalement pour créer une liaison à tolérance de perte et à faible latence entre différentes applications

### 2.9.4 SMTP (Simple Mail Transport Protocol)

Pour envoyer et distribuer des E-Mails sortants

### 2.9.5 FTP (File Transfer Protocol)

FTP permet aux utilisateurs de transférer des fichiers d'une machine à une autre. Les types de fichiers peuvent inclure des fichiers de programme, des fichiers multimédias, des fichiers texte et des documents, etc.

### 2.9.6 HTTP (HyperText Transfer Protocol)

HTTP est conçu pour transférer un hypertexte entre deux ou plusieurs systèmes. HTTP est un protocole de couche supérieure utilisé par les applications

### 2.9.7 ICMP (Internet Control Message Protocol)

permet les corrections d'erreurs de transferts

### 2.9.8 ARP (Address Resolution Protocol)

ARP est un protocole effectuant la convertir les adresses IP aux adresses de machines physiques (Adresse MAC) reconnues dans le réseau

### 2.9.9 SNMP (Simple Network Management Protocol)

SNMP est un protocole de couche application utilisé pour la gestion des périphériques réseau. Ce protocole peut collecter et manipuler des informations réseau précieuses à partir de commutateurs, de routeurs, de serveurs, d'imprimantes et d'autres périphériques connectés au

réseau

### 2.9.10 DNS (Domain Name System)

Le protocole DNS aide à convertir les noms d'hôte en adresses IP. Le serveur DNS est une base de données qui comprend le nom de domaine d'un site web, que les gens utilisent pour accéder au site web Le DNS est important car il peut rapidement fournir aux utilisateurs des informations, ainsi qu'un accès à des hôtes distants et à des ressources sur internet

### 2.9.11 DHCP (Dynamic Host Configuration Protocol)

DHCP est un protocole qui permet d'attribuer dynamiquement la configuration des hôtes, il permet d'éviter les problèmes et des erreurs de saisie manuelle et réduit les charges administratives DHCP fonctionne sur un modèle client-serveur

## 2.10 Conclusion

Dans ce chapitre nous avons présenter les notions et concepts de base sur les réseaux informatiques, à savoir le protocole de communication, topologie, architecture etc...

Le chapitre suivant présente les concepts et principe de base de protocole de routage et expliquer les principaux outils et techniques utilisées dans ce contexte

# Chapitre 3

# Protocole de routage

## 3.1 Introduction

Le travail d'un réseau consiste essentiellement à trouver les bons chemins pour un chaque paquet de sa source vers sa destination.

Le routage est le processus utilisé par votre ordinateur pour transmettre un paquet entre différents sous-réseaux. Si vous souhaitez communiquer avec un ordinateur sur un sous-réseau différent du vôtre, votre ordinateur doit transmettre les paquets de données à un routeur. Un routeur est le logiciel et le matériel responsables de la livraison des paquets entre deux sous-réseaux. Chaque routeur utilise une table de routage interne pour déterminer le meilleur chemin pour envoyer un paquet [6].

Le rôle décisif qui joue le routage et l'interconnexion complexe des réseaux internet font de la conception des protocoles de routage un défi majeur que doivent relever les développeurs de logiciels réseau. Par conséquent, la plupart des études relatives au routage concerne la conception des protocoles, très peut traiter de la configuration correcte des protocoles de routage. Toutefois, nombre de problèmes quotidiens résultent plutôt d'une mauvaise configuration des routeurs utilisés que de l'emploi d'algorithme mal conçus. C'est le rôle l'administrateur réseau de s'assurer que la configuration du routage est correcte.

## 3.2 Table de routage

Une table de routage est un ensemble de règles, souvent affichées sous forme de tableau, qui est utilisé pour déterminer où les paquets de données circulant sur un réseau IP (Internet Protocol) seront dirigés. Tous les appareils compatibles IP, y compris les routeurs et les commutateurs, utilisent des tables de routage.

Une table de routage contient les informations nécessaires pour transmettre un paquet le long du meilleur chemin vers sa destination. Chaque paquet contient des informations sur son origine et sa destination. Lorsqu'un paquet est reçu, un périphérique réseau examine le paquet et le fait correspondre à l'entrée de la table de routage fournissant la meilleure correspondance pour sa destination. Le tableau fournit ensuite au périphérique des instructions pour envoyer le paquet au saut suivant sur son itinéraire à travers le réseau.

Une table de routage est constituée des éléments suivants

- Méthode de routage : type de protocole qui a appris la route
- Réseau et masque : destination
- Distance administrative : préférence d'une route par un protocole sur un autre. Chaque protocole a sa valeur par défaut
- Valeur de métrique : valeur d'une route sur une autre parmi toutes celles apprises par un protocole de routage
- Via prochaine interface (Gateway)
- Interface de la sortie du routeur

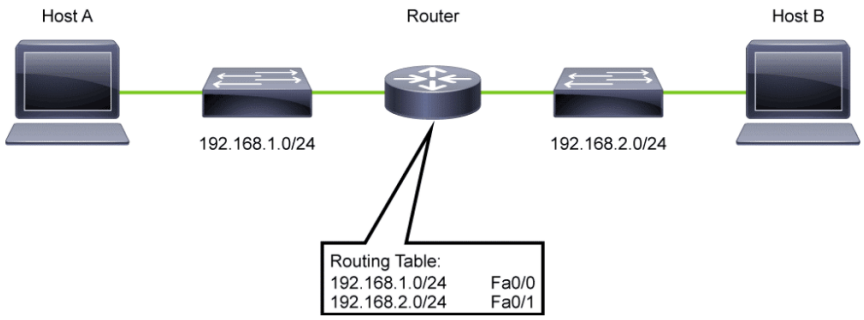



Figure 3.1 :Exemple d'une table de routage

## 3.3 Type de routage

### 3.3.1 Le routage statique

Le terme routage statique désigne l'utilisation de routes statiques configurées ou injectées manuellement à des fins de transfert de trafic. Ces routes statiques sont utilisées pour des raisons de sécurité en cas de dysfonctionnement.

Les opérations de routage statique s'articulent comme suit :

- L'administrateur réseau configure la route
- Le routeur insère la route dans la table de routage
- Les paquets sont acheminés à l'aide de la route statique

### 3.3.2 Le routage Dynamique

Le routage dynamique permet au réseau de s'adapter automatiquement aux modifications de la topologie, sans intervention de l'administrateur. Une route statique ne peut pas répondre dynamiquement aux modifications du réseau.

Si une liaison échoue, la route statique n'est plus valide si elle est configurée pour utiliser cette liaison défaillante, une nouvelle route statique doit donc être configurée. Si un nouveau routeur ou un nouveau lien est ajouté, ces informations doivent également être configuré sur chaque routeur du réseau. Dans un réseau très étendu ou instable, ces changements peuvent entrainer un travail considérable pour les administrateurs réseau. Cela peut aussi prendre beaucoup de temps pour chaque routeur[8].

Dans le réseau pour recevoir les informations correctes. Dans de telles situations, il peut être préférable que les routeurs reçoivent des informations sur les réseaux et les liaisons les uns des autres à l'aide d'un protocole de routage dynamique.

La fonction des protocoles de routage dynamique inclut les éléments suivants :

- Découverte des réseaux distants

- Actualisation des informations de routage
- Choix du meilleur chemin vers les réseaux de destination
- Capacité à trouver un nouveau meilleur chemin si le chemin actuel n'est plus disponible

## 3.4 Protocole de routage

Un protocole de routage spécifie comment les routeurs communiquent entre eux, diffusant des informations qui leur permettent de sélectionner des itinéraires entre deux nœuds quelconques sur un réseau informatique. Les algorithmes de routage déterminent le choix spécifique de la route. Chaque routeur n'a connaissance a priori que des réseaux qui lui sont directement rattachés. Un protocole de routage partage d'abord ces informations entre les voisins immédiats, puis sur l'ensemble du réseau. De cette façon, les routeurs acquièrent une connaissance de la topologie du réseau [6].

Il existe 2 types de protocoles de routage dynamique, ceux-ci diffèrent par la manière dont ils découvrent et effectuent des calculs sur les itinéraires :

- Protocoles de vecteur de distance (Distance Vector Protocol)
- Protocoles d'état de lien (Link State Protocol)

### 3.4.1 Protocoles de vecteur de distance

Les protocoles de routage à vecteur de distance sont des protocoles permettant de construire des tables de routages ou aucun routeur ne possède la vision globale du réseau, la diffusion des routes se faisant de proche en proche. Le terme (vecteur de distances) vient du fait que le protocole manipule des vecteurs de distances vers les autres nœuds du réseau. Des exemples de protocoles de vecteur de distance comprennent RIP, IGRP, EIGRP [8]

### 3.4.2 Protocoles à l'état de lien

Dans cette famille de protocoles chaque routeur communique avec tous les autres routeurs en échangent des informations permettant à chacun de construire une vue complète de la topologie du réseau ainsi qu'une table de routage, Prenant en compte les meilleures routes.

Cette méthode de routage permet une construction plus rapide des tables de routage que le routage a vecteur de distance. Des exemples de protocoles de vecteur de distance comprennent OSPF, ISIS

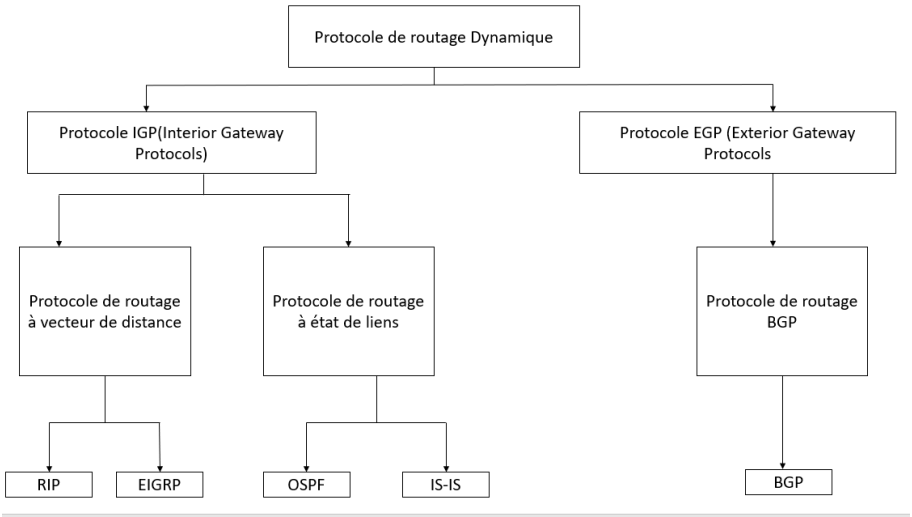



Figure 3.2 :Classification des protocoles de routage

## 3.5 Protocol OSPF (Open Shortest Path First)

### 3.5.1 Définition

OSPF (Open Shortest Path First) est un protocole de routage à état de liens qui a été développé comme alternative au protocole de routage à vecteur de distance, ou RIP.

Le protocole OSPF présente des avantages considérables par rapport au protocole RIP car il offre une convergence plus rapide et s'adapte mieux aux réseaux de plus grande taille.

OSPF est un protocole de routage à états de liens (Link State) qui prend en charge le concept de zones pour assurer l'évolutivité. Un administrateur réseau peut diviser le domaine de routage en zones distinctes qui aident à contrôler le trafic de mise à jour de routage.

Le routeur crée la table topologie à l'aide des résultats des calculs basés sur l'algorithme SPF de Dijkstra

### 3.5.2 Les Type de paquet OSPF

OSPF contient 5 types de LSPs (Link State paquets)

• Hello :
Utilisé pour établir et maintenir le voisinage. Figure 3.3

• DBR (Database Description) :
Liste résumée de la base de données de l'émetteur

• LSR (Link-State Request) :
Utilisé pour demander plus d'informations sur une entrée dans le DBD

• LSU (Link-State Update) :
Informations sur l'états des liens

• LSAck (LSA Acknowledgment) :
Le routeur envoie un acquittement (LSAck) pour confirmer la réception d'un LSU


Figure 3.3 :Paquet Hello

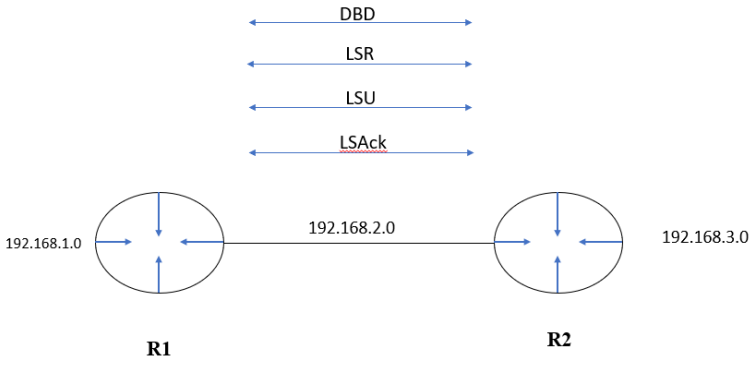



Figure 3.4 :Les Types de paquet OSPF

### 3.5.3 Paquet Hello OSPF

Les paquets Hello sont utilisés pour :

• Découvrit des voisins OSPF et établir des contiguïtés

• Annoncer les paramètres sur lesquels les deux routeurs doivent s'accorder pour devenir voisins

• Définir le routeur désigné (DR) et le routeur désigné de secours sur les réseaux à accès multiple, de type Ethernet et Frame Relay

### 3.5.4 Les tables de protocole OSPF

Chaque routeur OSPF stocke les informations de routage et de topologie dans trois tables :

1. Table de voisins (Neighbor table) :

Table de voisinage contient des informations sur le directement voisins OSPF connectés formant une contiguïté.

2. Table de Base de données (DataBase table) :

Table de base de données contient des informations sur l'ensembles vue de la topologie par rapport à chaque routeur.

3. Table de routage (Routing table) :

Table de routage contient des informations sur meilleur chemin calculé par l'algorithme de chemin le plus court en premier dans la table de la base de données.

## 3.6 Configuration basique de protocole OSPF

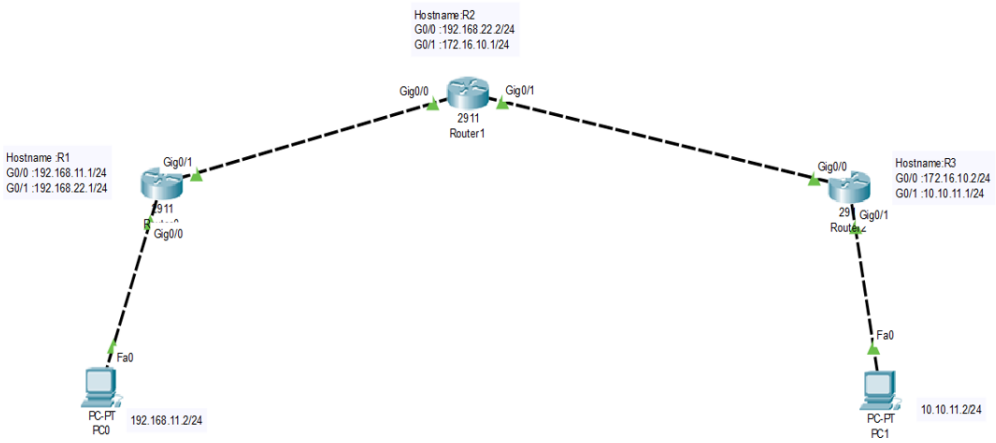



Figure 3.5 :Exemple de Configuration

### 3.6.1 Configuration R1

```
R1 (config)♯ router ospf 10

R1(config-router) ♯ router-id 1.1.1.1

R1(config-router) ♯ interface g0/0

R1(config-if) ♯ ip ospf 10 area 1

R1(config-router) ♯ interface g0/1

R1(config-if) ♯ ip ospf 10 area 0

R1(config-if) ♯ ex
```

### 3.6.2 Configuration R2

```
R2 (config) ♯ router ospf 10

R2(config-router) ♯ router-id 2.2.2.2

R2(config-router) ♯ interface g0/0

R2(config-if) ♯ ip ospf 10 area 0

R2(config-router) ♯ interface g0/1

R2(config-if) ♯ ip ospf 10 area 0

R2(config-if) ♯ ex
```

### 3.6.3 Configuration R3

```
R3 (config) ♯ router ospf 10

R3(config-router) ♯ router-id 3.3.3.3

R3(config-router) ♯interface g0/1

R3(config-if) ♯ip ospf 10 area 2

R3(config-router) ♯interface g0/0

R3(config-if) ♯ip ospf 10 area 0

R3(config-if) ♯ex
```

## 3.7 Vérification

### 3.7.1 Vérification des voisins OSPF

Affiche une liste détaillée des voisins, leur priorité et leur statut. Show ip ospf neighbor

```
Router>en
Router#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           1   FULL/DR         00:00:35    192.168.22.1    GigabitEthernet0/0
3.3.3.3           1   FULL/DR         00:00:36    172.16.10.2     GigabitEthernet0/1
Router#
```

### 3.7.2 Vérification DataBase

Affiche le contenu de la base de données topologique (router-Id, process-Id). Show ip ospf DataBase

```
R1#show ip ospf database
            OSPF Router with ID (1.1.1.1) (Process ID 10)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.1         1.1.1.1         298         0x80000005 0x0053f3 1
3.3.3.3         3.3.3.3         287         0x80000005 0x007432 1
2.2.2.2         2.2.2.2         287         0x80000006 0x00adef 2

                Net Link States (Area 0)
Link ID         ADV Router      Age         Seq#       Checksum
192.168.22.1    1.1.1.1         298         0x80000003 0x00aaa6
172.16.10.2     3.3.3.3         287         0x80000003 0x0053a8

                Summary Net Link States (Area 0)
Link ID         ADV Router      Age         Seq#       Checksum
192.168.11.0    1.1.1.1         477         0x80000003 0x00469d
10.10.11.0      3.3.3.3         366         0x80000003 0x00c070

                Router Link States (Area 1)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.1         1.1.1.1         459         0x80000004 0x00993a 1

                Summary Net Link States (Area 1)
Link ID         ADV Router      Age         Seq#       Checksum
192.168.22.0    1.1.1.1         476         0x80000007 0x00c410
172.16.10.0     1.1.1.1         282         0x80000008 0x007d0e
10.10.11.0      1.1.1.1         272         0x80000009 0x00052c
R1#
```

### 3.7.3 Vérification ip route

```
Router#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/24 is subnetted, 1 subnets
O IA    10.10.11.0/24 [110/2] via 172.16.10.2, 01:10:12, GigabitEthernet0/1
     172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C       172.16.10.0/24 is directly connected, GigabitEthernet0/1
L       172.16.10.1/32 is directly connected, GigabitEthernet0/1
O IA 192.168.11.0/24 [110/2] via 192.168.22.1, 01:10:22, GigabitEthernet0/0
     192.168.22.0/24 is variably subnetted, 2 subnets, 2 masks
C       192.168.22.0/24 is directly connected, GigabitEthernet0/0
L       192.168.22.2/32 is directly connected, GigabitEthernet0/0

Router#
```

### 3.7.4 Vérification Ping

```
Router#ping 10.10.11.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.10.11.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms

Router#
```

```
C:\>ping 10.10.11.2

Pinging 10.10.11.2 with 32 bytes of data:

Reply from 10.10.11.2: bytes=32 time<1ms TTL=125
Reply from 10.10.11.2: bytes=32 time<1ms TTL=125
Reply from 10.10.11.2: bytes=32 time<1ms TTL=125
Reply from 10.10.11.2: bytes=32 time<1ms TTL=125

Ping statistics for 10.10.11.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

```
Packet Tracer PC Command Line 1.0
C:\>ping 192.168.11.2

Pinging 192.168.11.2 with 32 bytes of data:

Reply from 192.168.11.2: bytes=32 time<1ms TTL=125
Reply from 192.168.11.2: bytes=32 time=11ms TTL=125
Reply from 192.168.11.2: bytes=32 time=18ms TTL=125
Reply from 192.168.11.2: bytes=32 time<1ms TTL=125

Ping statistics for 192.168.11.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 18ms, Average = 7ms

C:\>
```

## 3.8 Conclusion

Les routeurs permettant de choisir le chemin que les données vont emprunter pour arriver jusqu'à la destination.

Un protocole de routage est un ensemble de processus d'algorithme et de message qui permet d'échanger des informations sur les différents itinéraires, les informations de routage sont mémorisées dans la table de routage des équipements, cette table doit être périodiquement mise

à jour :

- Manuellement : Routage statique
- Automatiquement : Routage dynamique

Le but du routage est donc d'affirmer qu'il existe toujours un chemin pour aller d'un réseau à un autre.

# Chapitre 4

# Algorithmes évolutionnaires

## 4.1 Introduction

L'optimisation est une branche des mathématiques et de l'informatique, et de nombreuses recherches, à la fois pratiques et théoriques. Il existe deux grandes approches de l'optimisation. La première approche est dite déterministe : Les algorithmes de recherche utilisent toujours le même cheminement pour arriver à la solution. Les méthodes déterministes sont généralement efficaces quand l'évaluation de la fonction est très rapide, l'autre est aléatoire (non déterministe), l'algorithme ne suivre pas le même cheminement pour aller vers la solution. C'est la seconde approche, que va s'orienter notre travail vers un type de l'algorithme bien précis, les algorithmes du type évolutionnaires [29].

Ce chapitre présente succinctement la notion des algorithmes évolutionnaires d'une manière générale, Les algorithmes évolutionnaires (AEs) sont des méthodes de recherche inspirées par la théorie darwinienne de l'évolution. Bases sur une simulation d'évolution de populations de solutions.

## 4.2 Les Algorithmes évolutionnaires (AE)

La naissance des algorithmes évolutionnaires marquée dans les années 60 et 70. Les algorithmes évolutionnaires sont des méthodes stochastiques d'optimisation globale basées sur la théorie Darwinienne de l'évolution [34].

Les AE sont inspirés du concept de sélection naturelle élaboré par Charles Darwin. Le vocabulaire employé est directement calqué sur celui de la théorie de l'évolution et de la Génétique. Ils sont introduits par Fogel en 1965, Rechenberg en 1973, Holland en 1975 et popularisés par Goldberg en 1989.L'objectif étant d'arriver à des actions optimales pour atteindre les résultats

souhaités.

### 4.2.1 Le squelette

Les algorithmes évolutionnaires sont basés sur des principes simples. En effet, ils font évoluer une population d'individus, dont l'objectif de trouver la solution optimale, ou seulement les mieux adaptés à un environnement donné peuvent survivre et se reproduire [32].

Ces algorithmes reposent sur une vision darwinienne relativement simpliste et une optimisation stochastique résumée dans le diagramme de la figure 4.1

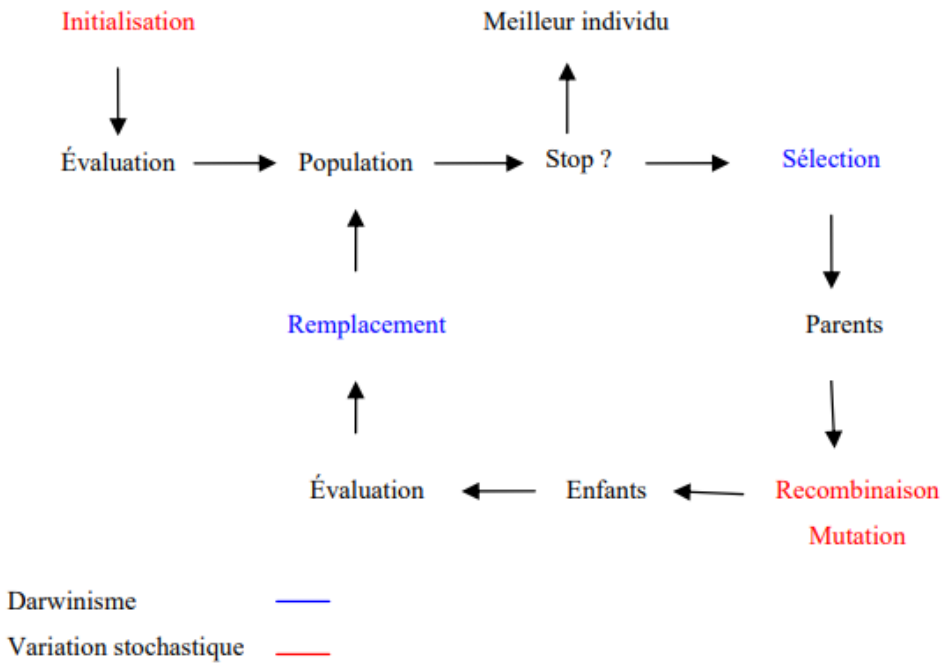



Figure 4.1 :Notions de base de l'algorithme évolutionnaire

### 4.2.2 Catégories des algorithmes évolutionnaires

On distingue quatre grandes familles historiques d'algorithmes évolutionnaires comme indiqué par la figure 4.2. Chacun de ces algorithmes manipule une population dont la taille, contrairement à la population naturelle [32].

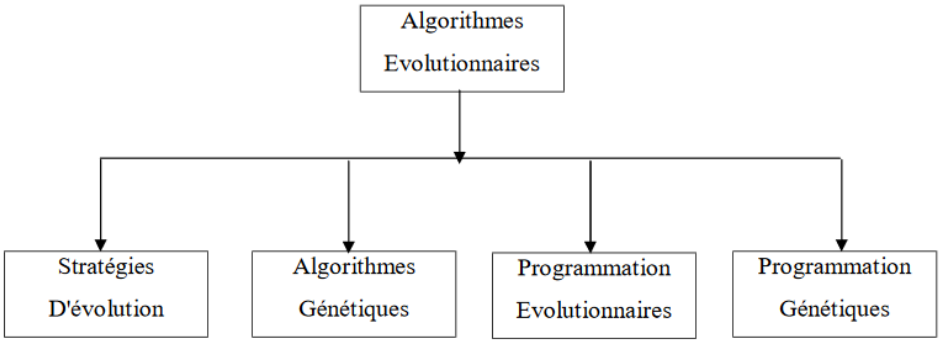



Figure 4.2 :Différentes classes d'algorithmes évolutionnaires

### Algorithme génétique (AGs)

Développés par (J. Holland 1975, Goldberg 1989, Michigan 1992),les AG ont été imaginés comme outils de modélisation de l'adaptation. Ces algorithmes sont les plus connus et utilisés des algorithmes évolutionnaires. Ils sont inspirés des mécanismes de la génétique et de l'évolution naturelle.

### Stratégies d'évolution (SE)

Développés par (I. Rechenberg et H.P. Schwefel, 1965),elles ont été développées pour résoudre des problèmes d'optimisation à variables. Ce sont les meilleurs algorithmes pour les problèmes purement numérique.

### Programmation évolutive (PE)

Développés par (L. J. Fogel,1966 et D.B. Fogel,1991,1995),les principales caractéristiques de ces algorithmes par rapport aux autres algorithmes évolutionnaires, à l'origine, le fait qu'ils n'utilisaient pas des individus par croisement et surtout l'utilisation d'un mode de sélection originale, mais ce modèle évolutionnaire faire allusion l'utilisation de la mutation.

### Programmation génétique (PG)

Développés par (J. Koza,1990), California,le modèle PG s'inspire des algorithmes génétiques, a pour but de créer automatiquement un programme pour résoudre un problème. Pour cela on va utiliser un petit programme qui vont évoluer et s'assembler pour fournir un programme plus complet. Ces programmes sont en général des S-expressions LISP, avec l'avantage facilement

représentés par des arbres syntaxiques.

## 4.3 Algorithmes génétiques

### 4.3.1 Définition

Les algorithmes génétiques (AGs), ont été développés par John Holland (1975), inspirées des mécanismes de l'évolution darwinienne (sélection naturelle) et de la génétique. Ces algorithmes font partie de la classe des algorithmes dits stochastiques. Les algorithmes génétiques permettant à une population de solution vers les solutions optimales. Pour ce faire, ils vont utiliser un mécanisme de sélection des individus de la population.

Les Algorithmes génétique (AGs) utilisent deux stratégies importantes pour trouver une solution ou un ensemble de solutions. Ces stratégies sont : l'exploration et l'exploitation. Ces mécanismes d'exploration et l'exploitation vont permettre de converger vers les bonnes solutions

### 4.3.2 Terminologie

Avant d'aborder comment fonctionnent les AGs, nous devons également définir la terminologie employée. La figure 03 présente une récapitulation de la terminologie naturelle et celle utilisée par les algorithmes génétiques.

| **Nature** | **Algorithme génétique** |
|---|---|
| Chromosome | Chaine |
| Gène | Trait, caractéristique |
| Allèle | Valeur de la caractéristique |
| Locus | Position dans la chaine |
| Génotype | Ensemble des valeurs des gènes |
| Phénotype | Ensemble de paramètres , structure décodée<br>Evaluation d'un génotype |

Figure 4.3 :La terminologie Naturelle et celle des algorithmes génétiques

### 4.3.3 Principe de fonctionnement des AGs

Un algorithme génétique fonctionne typiquement à travers un cycle simple de quatre étapes [34] :

- Création d'une population de chromosomes
- Évaluation de chaque chromosome
- Sélection des meilleurs chromosomes
- Manipulation génétique, pour créer une nouvelle population de chromosomes

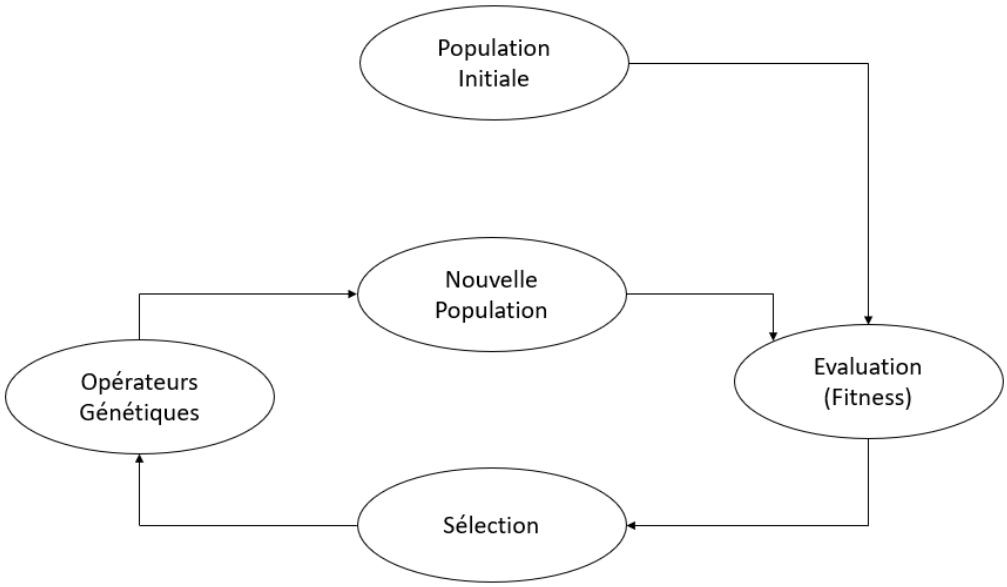



Figure 4.4 : Cycle Génétique

Le cycle décrit par la figure 4.4 est inspiré par la terminologie génétique. Initialement, une population initiale est générée ou chaque individu solution de la population est codé sous forme d'une chaine de caractères (chromosomes). Ensuite, une évaluation de chaque chromosome sera établie. Cette évaluation consiste à évaluer la qualité des chromosomes à l'aide de la fonction d'évaluation : Fitness. Ce qui permet de sélectionner les chromosomes les plus adaptés et par conséquent leur appliquer les opérateurs génétiques (croisement et mutation) ce qui crée une nouvelle génération.

A la fin du cycle, une nouvelle population est acquise ouvrant ainsi la voie pour une nouvelle génération et par conséquent un nouveau cycle.

## 4.4 Les caractéristiques des AGs

### 4.4.1 Codage

Le codage est une partie très importante des algorithmes génétiques. Le codage est une modélisation d'une solution d'un problème donné sous forme d'une séquence de caractères appelés chromosome, dit aussi gène, représente une variable ou une partie du problème. La tâche principale consiste à choisir le contenu des gènes qui facilite la description du problème et respecte ses contraintes. Plusieurs types de codages sont utilisés, on citera à titre d'exemple : codage réel, codage binaire, Gray.

#### Codage binaire

Goldberg et Holland ont démontré qu'il est idéal de représenter le chromosome en une chaine binaire. Un chromosome sera donc un ensemble de gènes. Dans le codage binaire le gène est codé par un caractère binaire 0 ou 1. C'est le plus courant et celui qui a été employé lors de la première application des algorithmes génétiques. Le codage binaire est également indépendant des opérateurs génétiques (croisement et mutation). Dans la pratique, le codage binaire peut présenter des difficultés, il est parfois très difficile de coder des solutions de cette manière.

#### Codage de Gray

Le code de Gray, également appelé code Gray ou code binaire réfléchi, Dans ces codes lors du passage d'une étape à une autre, seul un bit du groupe de codes change. Cette propriété est importante pour plusieurs applications.

Le codage de Gray est un codage qui a comme propriété qu'entre un élément n et un élément n+1, donc voisin dans l'espace de recherche, un seul bit diffère

Exemple :

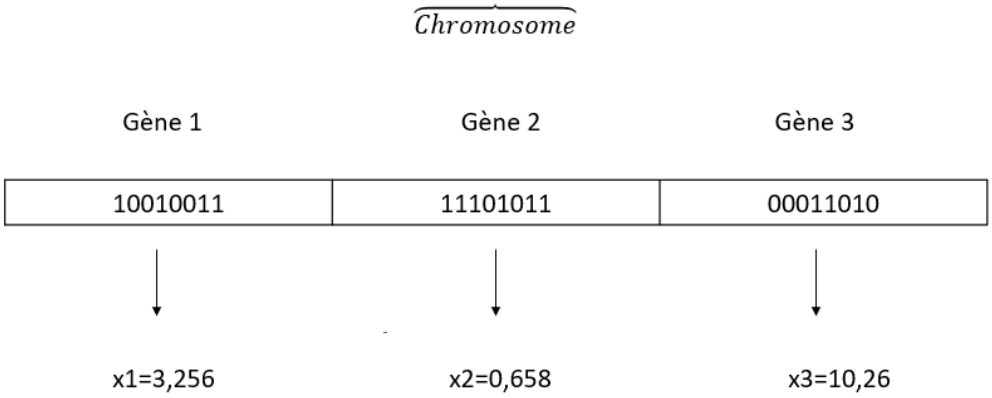
| Codage Décimale | Codage Binaire | Codage de Gray |
|---|---|---|
| 0 | 0000 | 0000 |
| 1 | 000**1** | 000**1** |
| 2 | 00**10** | 001**1** |
| 3 | 001**1** | 001**0** |
| 4 | 0**100** | 0**11**0 |

**Codage Réel**

David, Janikow et Michalewicz ont effectué une comparaison entre la représentation binaire et la représentation réelle. Ces autres ont trouvé que la représentation réelle donne de meilleurs résultats d'après leur problème à résoudre.

Un gène est ainsi représenté par un nombre réel au lieu d'avoir à coder les réels en binaire. Le codage réel permet d'augmenter l'efficacité de l'algorithme génétique. En effet, un chromosome code en réels est plus court que celui codé en binaire.

Exemple :




### 4.4.2 Évaluation (Fitness)

Pour mesurer les performances de chaque individu qui correspond à une solution donnée du problème à résoudre on introduit une fonction d'évaluation appelé Fitness. Elle permet d'évaluer les solutions optimales.

La fonction de fitness simplement définie est une fonction qui prend une solution candidate au problème en entrée et produit en sortie à quel point la solution est « adaptée » ou « bonne » par rapport au problème considéré._

Le calcul de la valeur de fitness est effectué à plusieurs reprises dans un AG et doit donc être suffisamment rapide.

Ces deux derniers éléments, codage et évaluation, sont les seuls éléments spécifiques au problème à résoudre. Une fois qu'ils sont fixés, l'algorithme génétique que l'on appliquera sera toujours le même.

$$f(x) = \sum_{i=1}^{i=k} d_{ij} \qquad (1)$$

.

Le fitness peut être une probabilité sur la valeur de la fonction objectif

$$p(i) = \frac{f(i)}{\sum_{k=1}^{N} f(k)} \qquad (2)$$

## 4.5 Les opérateurs génétiques

Une fois la génération initiale créer, l'algorithme fait évoluer la génération à l'aida des opérateurs suivants. Ces opérateurs se trouvent dans la littérature sous plusieurs variantes.

### 4.5.1 Opérateur de sélection

L'opérateur de sélection est chargé de favoriser les meilleurs individus. La sélection est fondée sur la qualité des individus, estimée à l'aide de fonction d'adaptation. Cet opérateur est permet de définir quels seront les individus de P qui vont être dupliqués dans la nouvelle population P' et vont servir de parents. Il existe plusieurs méthodes de sélection :

- Sélection par Rang (Ranking)
- Sélection par Roulette
- Sélection par Tournoi
- Sélection Uniforme

#### Sélection par rang

Cette méthode consiste à ranger d'abord les individus selon un ordre (croisement ou décroisement de leurs performances) puis à attribuer une probabilité de sélection en fonction de rang.

Cette probabilité est proportionnelle à la performance et donnée la formule suivante :

$$P_{sel}(xi) = \frac{Rang(xi)}{\sum_{i=1}^{PopS} Rang(xi)}$$

Exemple :

| Individual | Fitness | Rank | Probability |
|---|---|---|---|
| A | 8,2 | 6 | 28,57 % |
| B | 3,2 | 4 | 19,04 % |
| C | 1,4 | 3 | 14,2 % |
| D | 1,2 | 2 | 9,5% |
| E | 4,2 | 5 | 23,8 % |
| F | 0,3 | 1 | 4,7 % |
| Sum | 18,5 | 21 | 100% |

**Sélection par Roulette**

Cette méthode est la plus connue et la plus utilisée. Cette méthode consiste à sélectionner les individus, donc plus les individus sont adaptés au problème, plus ils ont de chances d'être sélectionnés. La probabilité de s élection est calculée à partir de la valeur de Fitness du chromosome.

Dans une population de N individus, alors la probabilité de sélection d'un individu $x_i$ notée$P(x_i)$ est égale à :

$$P_s(xi) = \frac{Fitness(xi)}{\sum_{i=1}^{N} Fitness(xi)}$$

Exemple :

| Individual | Fitness | Probability |
|---|---|---|
| A | 8,2 | 44,32 % |
| B | 3,2 | 17,30 % |
| C | 1,4 | 7,57 % |
| D | 1,2 | 6,49 % |
| E | 4,2 | 22,70 % |
| F | 0,3 | 1,62 % |
| Sum | 18,5 | 100% |

**Sélection par Tournoi**

Cette méthode est celle avec laquelle on obtient les résultats les plus satisfaisants. Elle consiste à choisir aléatoirement deux ou plusieurs individus et à sélectionner le plus fort. Ce processus est répété plusieurs fois jusqu'à l'obtenir de N individus.

La sélection des tournois, en général, tend à sélectionner les individus les plus aptes à la reproduction, mais un élément de probabilité garantit que les deux concurrents ont une chance.

**Sélection Uniforme**

C'est une technique très simple qui consiste à sélectionner un individu aléatoirement de la population P. La probabilité $P_{i pour qu'un individu soit slectionn est d finie par}$ :

$$P(i) = \frac{1}{TaillePop}$$

### 4.5.2 Opérateur de croisement (Crossover)

L'un des aspects uniques du travail impliquant des algorithmes génétiques est le rôle important que joue le croisement (recombinaison) dans la conception est la mise en œuvre de systèmes évolutifs robustes. Dans la plupart des AGs, les individus sont représentés par des chaines de longueur fixe et le croisement opère sur des paires d'individus (parents) pour produire de nouvelles chaines (offspring) en échangeant des segments des parents.

De nombreux types de croisement existent dans la littérature. Ils préservent plus ou moins l'identité génétique des parents et permettent un déplacement dans tout l'espace des solutions le type de croisement le plus simple est le croisement à un site [32].

### Croisement à un site

Une méthode couramment utilisée pour le croisement est appelée croisement à point unique. Dans cette méthode, une position de croisement à point unique (appelée point de coupure) est choisie au hasard et les parties de deux parents après la position de croisement sont échangées à partir de deux offspring, comme indiqué dans la figure 4.5 :

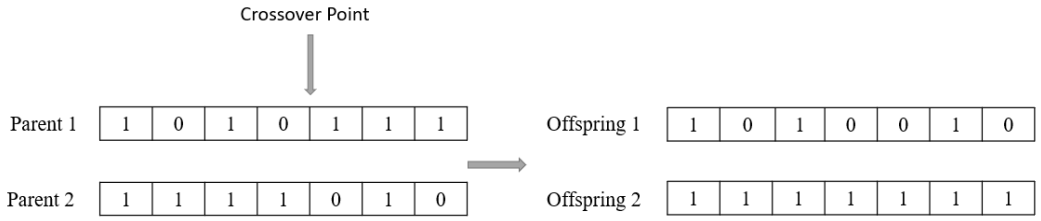



Figure 4.5 : Croisement à un site

### Croisement à K sites

Le croisement multipoint est une généralisation du croisement à points unique. Introduisant un nombre plus élevé de points de coupure. Dans ce cas, plusieurs positions sont choisies au hasard et les segments entre eux sont échangés, comme indiqué la figure suivante :

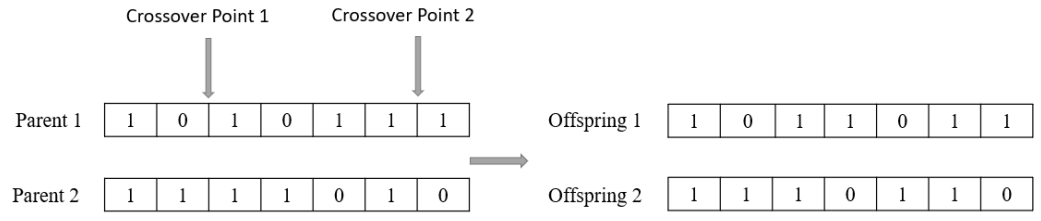



Figure 4.6 : Croisement à K sites

### Croisement Uniforme

Cette technique génère des progénitures gène par gène à partir des deux parents. La plus connue est celle qui utilise un masque. S'il est égal à 1, offspring 1 reçoit l'allèle correspondant du parant 1 et offspring 2 reçoit celui du parent 2. Sinon, l'échange se fait dans l'autre sens Figure 4.7 [35] :

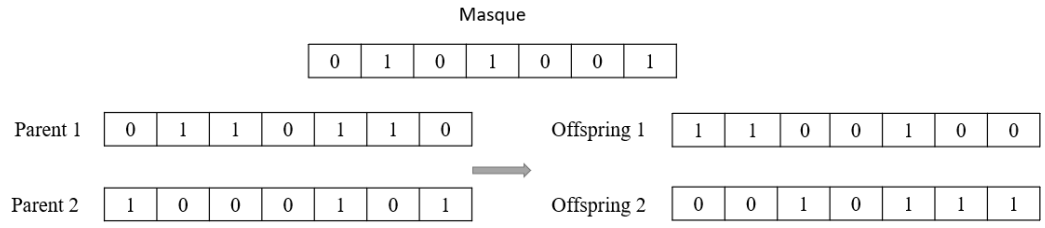



Figure 4.7 : Croisement Uniforme

### 4.5.3 Opérateurs de Mutation

Dans un Algorithme génétique simple, la mutation est définie étant la modification aléatoire d'une partie d'in chromosome. Lorsqu'un chromosome est choisi pour la mutation, une modification aléatoire est apportée aux valeurs de certains emplacements du chromosome. La mutation a pour rôle de maintenir une certaine diversité dans la population et protège les individus contre une perte des informations essentielle contenues dans leurs gènes [35].

Une méthode couramment utilisée pour la mutation est appelée mutation ponctuelle. À travers, un type de mutation spécial utilisé pour différents types de problèmes et méthodes de codage.

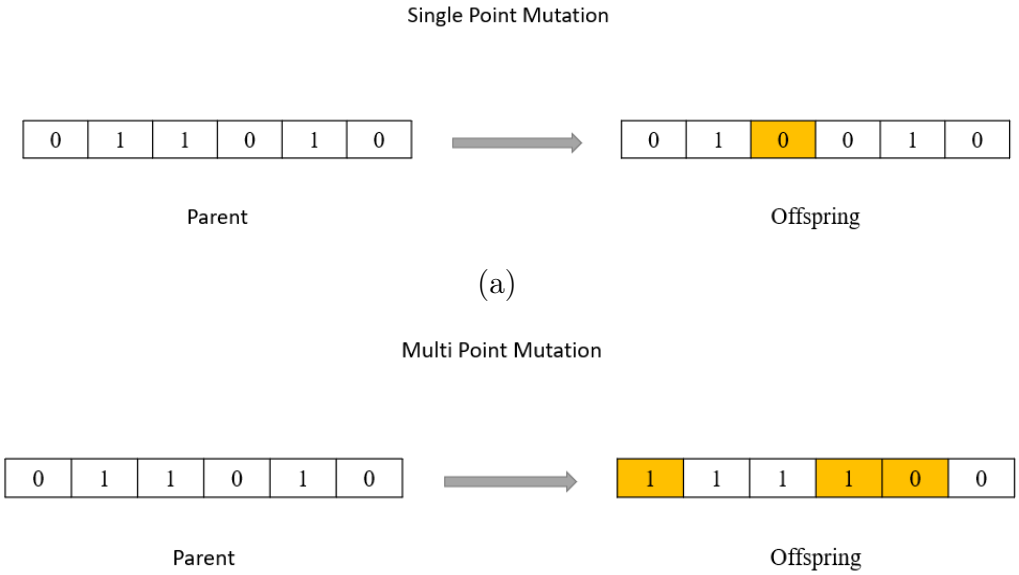



Figure 4.8 :Principe de l'opérateur de mutation

## 4.6 Conclusion

Dans cette partie, nous avons présenté les algorithmes évolutionnaires, nous avons introduit un rappel sur les fondations nécessaires des algorithmes évolutionnaires. Puis nous avons vue de manière générale les concepts des algorithmes génétique et donné les éléments nécessaires à la compréhension de la méthode. Ces algorithmes classés parmi les méthodes stochastiques, qui utilise précisément du principe de la sélection naturelle.

# Chapitre 5

# L'optimisation des poids pour le routage OSPF

## 5.1 Introduction

Le but de l'optimisation dans l'ingénierie du trafic est d'augmenter les performances du trafic IP en utilisant les ressources réseau de façon économique. Dans ce chapritre, comme complément aux traveaux de B. Fotz [26,27,28], nous décrivons une heuristique hybride basé sur les algorithmes génétiques afin de trouver un meilleur réglage des poids pour un routage OSPF.

## 5.2 Description du problème

### 5.2.1 Problème général du routage

Etant donné un graphe orienté pondéré G= (N,A,C), où les nœuds et les arcs représentent respectivement les routeurs et les liaisons les reliant. A chaque arc $u \in A$ du graphe G est associé un nombre C(u)>0 appelé capacité de l'arc u . On suppose, en outre, que l'on se donne une matrice de demande D, appelée aussi matrice de trafic. Chaque élément $d_{st} \in D$ correspond à une paire de nœuds $(s,t) \in N \times N$ et donne la quantité du trafic qu'on doit écouler du nœud s vers le nœud t où s représente le noeud source et t le nœud destination ou puits de la demande. Notons que plusieurs entrées de cette matrice peuvent être nulles et en particulier celles correspondantes aux pairs de nœuds qui ne sont relies par aucun chemin.

Pour un routage donné, la charge l(a) sur un arc $a \in A$ est le flot total a sur , c'est à dire le cumul des flots traversant l'arc a . L'utilisation d'une liaison a est le rapport $\frac{l(a)}{c(a)}$ .

L'objectif recherché est de garder les charges au-dessous des capacités, plus précisément, on doit distribuer les flots des demandes de telle sorte que la somme des coûts sur tous les

arcs soit minimale :

$$Min[\phi = \sum_{a \in A} \phi_a(\frac{l(a)}{c(a)})]$$

où le coût de l'arc a $\phi_a$ est fonction de l'utilisation.

Nous avons utilisé comme mesure de congestion la fonction continue par morceaux proposée dans [26] et reprise dans [28]. Plusieurs raisons nous ont poussés à utiliser cette fonction :

- reconnue comme une bonne approximation de la mesure de congestion

- acceptable par les solveurs de programmation linéaire

- facile à implémenter ;

- non gourmande en temps.

Nous voudrions comparer nos résultats avec celles données par les auteurs cités ci-dessus.

Cette fonction est définie comme suit :

A chaque arc $a \in A$,$\phi_a$ est la fonction de coût convexe linéaire par morceaux définie par (figure 5.1) :

$$\Phi_a = \begin{cases} l(a) & \text{pour } 0 \leq \frac{l(a)}{C(a)} \leq \frac{1}{3} \\ 3l(a) - \frac{2}{3}C(a) & \text{pour } \frac{1}{3} \leq \frac{l(a)}{C(a)} \leq \frac{2}{3} \\ 10l(a) - \frac{16}{3}C(a) & \text{pour } \frac{2}{3} \leq \frac{l(a)}{C(a)} \leq \frac{9}{10} \\ 70l(a) - \frac{178}{3}C(a) & \text{pour } \frac{9}{10} \leq \frac{l(a)}{C(a)} \leq 1 \\ 500l(a) - \frac{1468}{3}C(a) & \text{pour } 1 \leq \frac{l(a)}{C(a)} \leq \frac{11}{10} \\ 5000l(a) - \frac{16318}{3}C(a) & \text{pour } \frac{11}{10} \leq \frac{l(a)}{C(a)} \leq \end{cases}$$

A chaque arc $a \in A$ , $\phi_a$ est la fonction de cout convexe linéaire par morceaux définie par figure 5.1 :

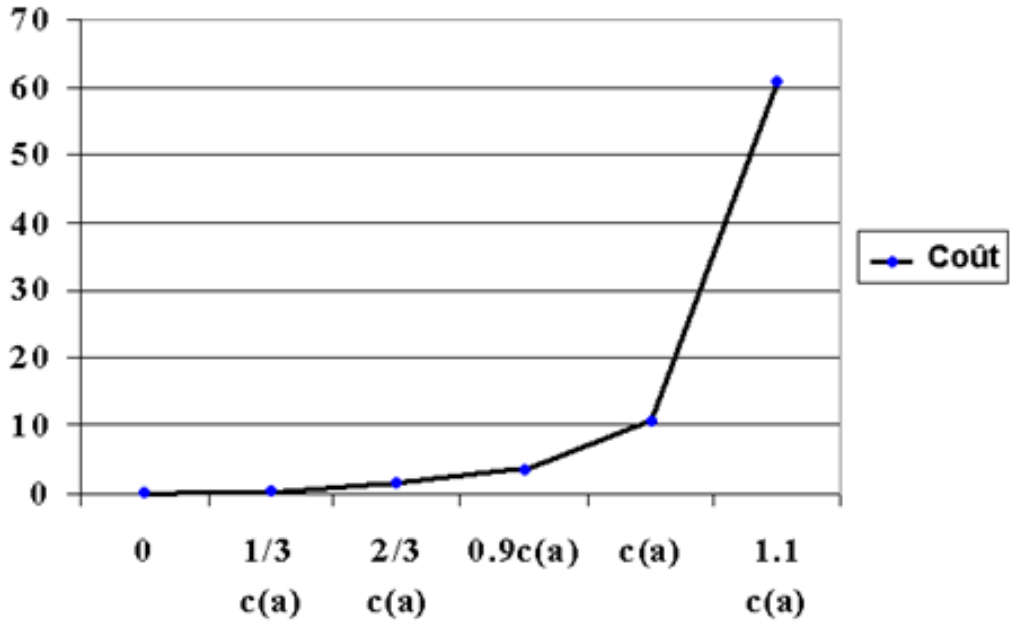



Figure 5.1 : Le cout d'arc $\phi_a(l(a))$ comme une fonction de la charge $l(a)$ Pour capacité C(a) =1

Le coût d'arc $\phi_a$ $(\frac{l(a)}{c(a)})$ est donc une fonction de l'utilisation $\frac{l(a)}{c(a)}$ . Elle modélise les délais causés par les paquets perdus. Tant que l'utilisation est inférieur à $\frac{1}{3}$ alors $\phi_a = 1$ .Le coût d'arc augmente jusqu'à la valeur 10,667 pour un arc à plein chage l(a)=C(a) , ensuite le coût augmente de façon exponentielle vers la valeur limite 5000.

Généralement, $\phi_a$ favorise l'envoi de flots sur les arcs avec de petite utilisation. Le coût croît progressivement quand l'utilisation s'approche de 100 , puis croît de façon exponentielle quand la capacité maximale est atteinte. Notre but est de maintenir aussi que possible l'utilisation maximale $\max_a \in A$ $\frac{l(a)}{c(a)} < 1$ .

Dans le problème général de routage, il n'y a pas de contrainte sur la façon de distribuer le flot le long des plus courts chemins. Ce problème peut être formulé et résolu dans un temps polynomial, comme un problème de multiflot.

Les flots qui traversent le réseau sont définis par la matrice de demandes. Cette dernière peut être construite à l'aides des mesures concrètes, ou par prédiction en se basant sur l'ensemble des contraintes des clients du réseau. Il est à noter que, puisque le trafic varie le jour et la nuit, on peut utiliser deux matrices distinctes.

Puisque le choix des poids d'arcs détermine les plus courts chemins, qui fournissent à leur tour le routage du trafic, les charges des arcs, et enfin la valeur de la fonction $\phi$ , alors on doit trouver, pour chaque arc a$\in A$ , un entier $W_a \in \{1,.....,65535\}$ qui minimise la fonction objectif $\phi$ . Forz et al[26] ont montré que ce problème est NP-difficile.

## 5.3 Formulation Mathématique

### 5.3.1 Routage Optimal

Rappelons qu'on nous donne un réseau orienté G = (N, A ,C) avec une capacité C(a) pour chaque $a \in A$ .De plus, nous avons une matrice de demande D qui pour chaque paire $(s,t) \in N \times N$ indique la demande D (s, t) dans le flux de trafic entre s et t. Nous appellerons parfois les entrées non nulles de D les demandes. A chaque couple (t) et à chaque arc a, on associe une variable $f_{(a)}^{(s,t)}$ indiquant quelle part du trafic de s à t passe par a. La variable $l(a)$ représente la charge totale sur l'arc $a$, c'est -à-dire la somme des flux passant par a , et $\phi_a$ , est utilisée pour modéliser la fonction de cout linéaire par morceaux de l'arc a.

Avec cette notation, le problème de routage général peut être formulé comme le programme linéaire suivant :

$$\min \left[ \Phi = \sum_{a \in A} \Phi_a(l(a)) \right]$$

Sous les contraintes :

$$\sum_{u:(u,v)\in A} f_{(u,v)}^{(s,t)} - \sum_{u:(v,u)\in A} f_{(v,u)}^{(s,t)} = \begin{cases} -d(s,t) & \text{si } v = s; \\ d(s,t) & \text{si } v = t; \\ 0 & \text{sinon} \end{cases} \qquad v,s,t \in N; \quad (2)$$

$$l(a) = \sum_{(s,t)\in NxN} f_a^{(s,t)} \qquad a \in A; \quad (3)$$

$$\Phi_a = \begin{cases} l(a) & \text{pour } 0 \le \frac{l(a)}{C(a)} \le \frac{1}{3} \\ 3l(a) - \frac{2}{3}C(a) & \text{pour } \frac{1}{3} \le \frac{l(a)}{C(a)} \le \frac{2}{3} \\ 10l(a) - \frac{16}{3}C(a) & \text{pour } \frac{2}{3} \le \frac{l(a)}{C(a)} \le \frac{9}{10} \\ 70l(a) - \frac{178}{3}C(a) & \text{pour } \frac{9}{10} \le \frac{l(a)}{C(a)} \le 1 \\ 500l(a) - \frac{1468}{3}C(a) & \text{pour } 1 \le \frac{l(a)}{C(a)} \le \frac{11}{10} \\ 5000l(a) - \frac{16318}{3}C(a) & \text{pour } \frac{11}{10} \le \frac{l(a)}{C(a)} \end{cases} \quad ;a \in A$$

(4)

$$f_{(a)}^{(s,t)} \geq 0\,; \qquad a \in A\,; \qquad s,t \in N.\ (5)$$

Les contraintes (2) sont les contraintes de conservation de flots qui assurent que le flux de trafic désiré est acheminé de s à t . Les contraintes (3) définissent la charge sur chaque arc. Les contrainte (4) définissent la fonction coût pour chaque arc.

Le programme ci-dessus est une formulation complète de programmation linéaire du problème général de routage

### 5.3.2 Routage OSPF

Dans le routage OSPF, nous choisissons un poids w(a) pour chaque arc $a \in A$ . La longueur d'un chemin est alors la somme de ses poids d'arcs, et nous avons la condition supplémentaire que tout le flux quittant un nœud destiné à une destination données est uniformément répartis sur des arcs sur les chemins les plus courts vers cette destination.

Plus précisément, pour chaque couple source destination $(s,t) \in N \times N$ , et pour chaque nœud $u \in A$ , on doit avoir :

$$\mathrm{f}_{(u,v)}^{(s,t)} \begin{cases} 0 & \text{si (u,v) n'est pas sur un plus court chemin de s vers t} \\ \mathrm{f}_{(u,v')}^{(s,t)} & \text{si (u,v) et (u,v') sont sur des plus courts chemins de s vers t} \end{cases} \quad (5.1)$$

### 5.3.3 Normalisation du coût

Nous utiliserons le facteur de normalisation d'échelle de la fonction coût proposée par fortz[26][] et qui nous permettra de comparer les coûts pour différents réseaux (taille et topologie). Pour définir cette mesure , on introduit la fonction :

$$\phi_{Uncap} = \sum_{(u,v)\in N\times N} d(u,v).dist_1(u,v)$$

où $dist_1(u,v)$ est la distance mesurée avec des poids unité (c'est à dire en nombre de sauts). $\phi_{Uncap}$ est alors la charge totale si tous les flots de trafic passent par les plus courts chemins de poids unité. La fonction de coût normalisée sera donc :

$$\phi^* = \frac{\phi}{\phi_{Uncap}}$$

Notons que si $\phi^* = 1$ , cela signifie que nous avons un routage le long des plus courts chemins avec toutes les charges restant inférieures à $\frac{1}{3}$ de la capacité. Dans cette situation idéale, il n'y a aucune raison d'augmenter les capacités du réseau. Si tous les arcs sont exactement à plein charge, le facteur coût est alors égal au seuil 10,667.

### 5.3.4 Evaluation du coût

Le problème de base est de calculer les charges résultant d'un réglage particulier de poids. En considérant une destination $t \in N$ à chaque fois, on peut calculer le flux total de toute source $t \in N$ à cette destination. Ceci donne lieu à une certaine charge partielle pour chaque arc $a \in A$ . Ayant trouvé toutes les charges partielles, on peut alors, calculer les charges sur tout arc $a \in A$

**Définition**

Un arc $(u, v) \in A$ est un SP-arc, si et seulement si, il se trouve sur un plus court chemin vers le puits $t \in N$ , c'est à dire, $d_u^t = w(u, v) + d_v^t$ .

Où $d_u^t$ est la plus courte distance de u vers t et (u,v)>0 est le poids de l'arc (u,v) .

**Définition**

Un sous-graphe de G, noté T=(N(T), A(T)) est un arbre du plus court chemin vers le puits, si et seulement si :

— T est un arbre orienté de racine .

— N(T) est l'ensemble de tous les sommets qui peuvent atteindre le puits dans G.

— chaque arc dans T est un SP-arc

Pour calculer le flot à t , on doit :

1. Utiliser l'algorithme de Dijkstra (par tas binaire) pour calculer toutes les distances $d_u^t$ à t . Normalement l'algorithme de Dijkstra calcule les distances à partir d'une même source. Il suffit donc de l'appliquer au graphe obtenu en inversant l'orientation de tous les arcs dans le graphe G.

2. Calculer l'ensemble $A^t$ de tous les arcs sur les plus courts chemins vers t, c'est à dire,le sous-graphe des plus courts chemins vers t :

$$A^t = \{(u,v) \in A\,;\ d_u^t - d_v^t = w_{(u,v)}\}$$

3. Calculer le degré sortant $\delta_u^t$ dans $A^t$ , pour chaque nœud u$\in t$ :

$$\delta_u^t = |\{v \in A;(u,v) \in A^t\}|$$

Cette valeur indiquera s'il y'a une répartition de charge au nœud u .

4. Calculer les charges partielles $l^t_{(v,w)}$ pour tout arc (v,w) sur les plus courts chemins vers t , c'est à dire, pour tout $(v,w) \in A^t$ .

Soit f le flux arrivant au nœud v , le flot sortant sur l'arc (v,w) est alors égal à $\frac{1}{\delta_v^t}(d(s,t)+f)$ car il y'a $\delta_v^t$ plus courts chemins de v à t . Calculons maintenant le flux f par figure 5.2

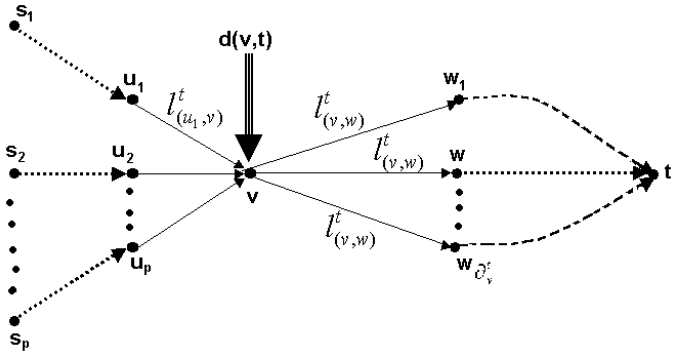



Figure 5.2 :Cas de répartition de charges.

Pour toute source $s \in N$ , $\sum_{(u,v)\in A^t} f^{(s,t)}_{(u,v)}$ est le flot arrivant à v sur les plus courts chemins depuis s . Le flot arrivant à sur tous les plus courts chemins est donc :

$$f = \sum_{s\in N} \sum_{(u,v)\in A^t} f^{(s,t)}_{(u,v)} = \sum_{(u,v)\in A^t} l^t_{(u,v)}$$

Où $l^t_{(u,v)}$ est la charge partielle sur l'arc (u,v) .

La charge partielle sur l'arc (v,w) est donc :

$$l^t_{(u,v)} = \frac{1}{\delta_v^t}\,[d(v,t) + \sum_{(u,v)\in A^t} l^t_{(u,v)}]$$

Les étapes précédentes du calcul des charges sont résumées dans l'algorithme de la figure 5.3 :

Remarque : Puisque le calcul de la charge partielle est récursive, les nœuds devront être visités dans l'ordre décroisant de la distance $d_v^t$ à t . Les étapes précédentes du calcul des charges sont résumées dans l'algorithme de la figure 5.3.

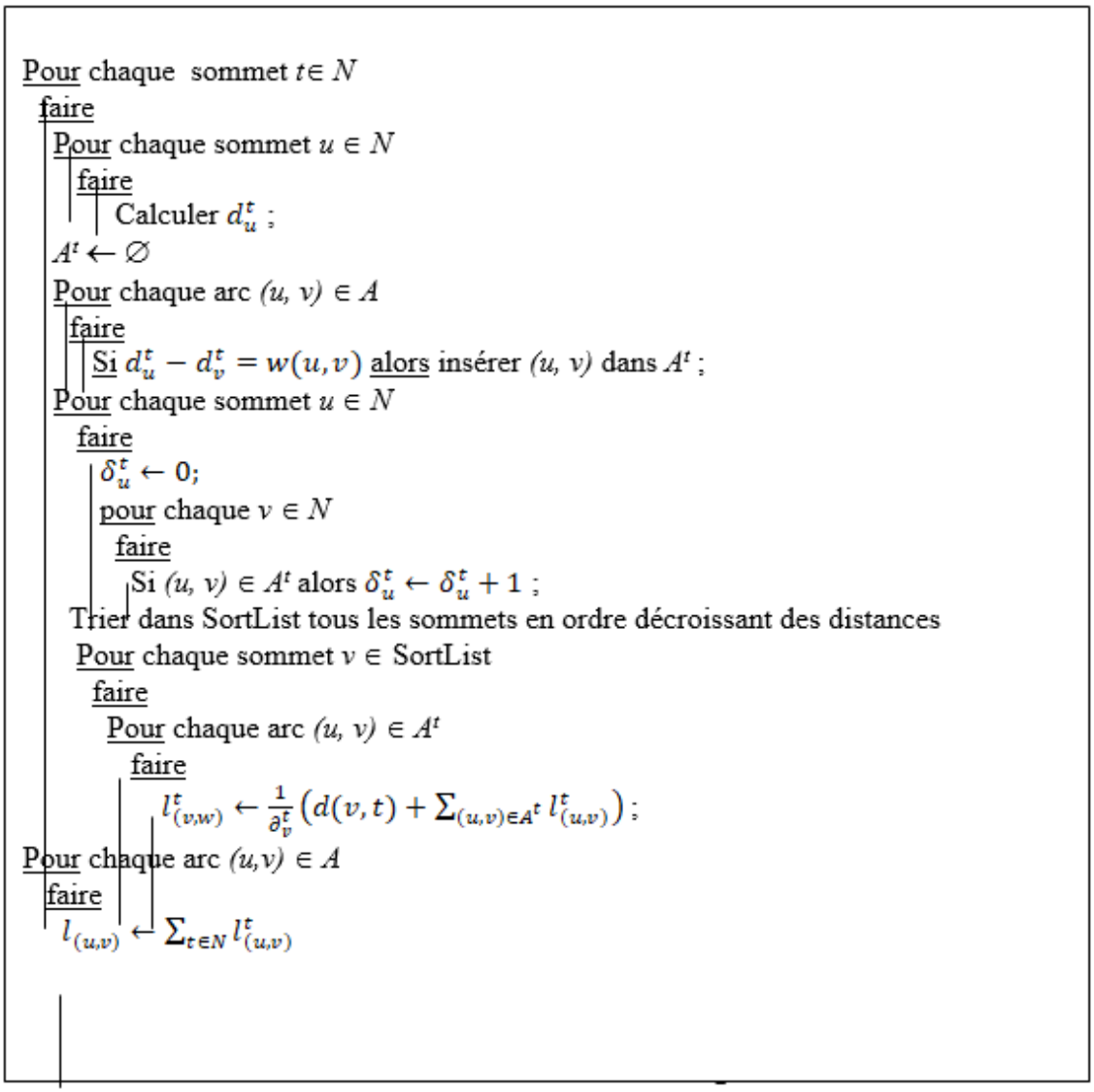
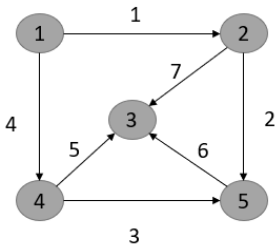



Figure 5.3

Exemple : Calculons les charges partielles et totales du graphe suivant :




| Arc | Origine | Extrémité | Poids | Capacité |
|---|---|---|---|---|
| 1 | 1 | 2 | 20 | 30 |
| 2 | 2 | 5 | 10 | 30 |
| 3 | 4 | 5 | 10 | 20 |
| 4 | 1 | 4 | 20 | 20 |
| 5 | 4 | 3 | 30 | 30 |
| 6 | 5 | 3 | 30 | 30 |
| 7 | 2 | 3 | 10 | 20 |

Figure 5.4 : Exemple de Graphe

En Appliquant l'algorithme ci-dessus à la matrice du trafic du tableau suivant, on obtient les charges suivantes :

| (u,v) | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | 0 | 10 | 5 | 10 | 5 |
| 2 | 0 | 0 | 2 | 0 | 2 |
| 3 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 5 | 0 | 5 |
| 5 | 0 | 0 | 2 | 0 | 2 |

Tableau : Matrice de Trafic

On remarque qu'il y'a sur le graphe deux plus courts chemins du nœud 1 vers le nœud 5

| t\a | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 10 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 5 | 0 | 0 | 0 | 5 | 2 | 7 |
| 4 | 0 | 0 | 0 | 10 | 0 | 0 | 0 |
| 5 | 2,5 | 4,5 | 7,5 | 2,5 | 0 | 0 | 0 |
| Charge | 17,5 | 4,5 | 7,5 | 12,5 | 5 | 2 | 7 |

Figure 5.5 : les charges des arcs 1 à 7

## 5.4 L'algorithme génétique

Nous étudions maintenant l'implantation de l'algorithme génétique publié par M. Ericsson et al[19] pour résoudre le problème de réglage de poids de routage OSPF. Dans tout ce qui suit cet algorithme sera désigné sous le nom GAOSPF.

### 5.4.1 Description sommaire de l'algorithme

Les algorithmes génétiques suivent tous un principe commun (chapitre 4) et l'implémentation présente n'y fait pas défaut : une phase de constitution de population initiale suivie de phases de régénération de populations grâce aux opérateurs génétiques de sélection, croisement et mutation. Chaque génération produira, en principe, des solutions plus performantes que les précédentes. Plus le nombre de générations est grand, plus la solution se raffinera et la dernière génération devrait contenir une bonne solution, mais qui n'est pas nécessairement optimale. Pendant le processus de création, deux populations existent en parallèle : la population courante, désigné par Pop, d'où sont puisés les chromosomes parents et la nouvelle génération, désignée par NewPop, en voie de réalisation, où se retrouveront les chromosomes enfants.

La première génération consiste en npop vecteurs de poids entiers générés aléatoirement w=$(w_1, ....., w_|A|)$ , avec $w_i \in W = \{1, ...., w_{maw}\}$ ; $i = 1....|A|$. Cette méthode a l'avantage de commencer la recherche à partir de diverses solutions de l'espace de recherche

#### Sélection des parents et reproduction

Le GAOSPF utilise une méthode de sélection obtenue en combinant plusieurs techniques connues dans la littérature des algorithmes génétiques et principalement le ranking. Les opérateurs génétique croisement et mutation ont un effet destructeur sur les individus de la génération suivante , il n'est donc pas possible de maintenir les individus les plus performants d'une génération à l'autre. C'est pourquoi un ajustement a été implémenté, il recopie systématiquement les meilleurs individus vers la génération suivante. Il améliore ainsi la convergence de l'algorithme tout en maintenant un parcours complet de l'espace des solutions exploré.

Le principe de cet algorithme est le suivant :

La population est triée par ordre décroissant de la valeur de la fonction coût $\phi$ , et les solutions sont alors classées dans trois catégories (figure ) :

1. La classe A : Classe des élites de taille $\alpha\%$ de npop ;

2. La classe C : Classe des plus mauvaises solutions de taille $\beta\%$ de npop ;
3. La classe B : Classe des chromosomes restants de taille $(100-(\alpha+\beta))\%$ de npop ;

Pendant la phase de reproduction, les chromosomes sont sélectionnés et leurs structures sont modifiées pour générer de nouveaux individus qui vont former la génération suivante. GAOSPF procède comme suit (figure ) :

•Appliquer a la classe A la stratégie élitiste qui favorise les meilleurs individus , c'est à dire, tous les individus de cette classe sont promu à la génération suivante sans aucun traitement ;

• Combiner un parent de la classe A avec un parent des classes B et C, en utilisant les opérateurs génétiques croisement et mutation afin de produire des enfants de la classe B de la génération suivante ;

• Enfin, la classe C de la génération suivante est à nouveau générée aléatoirement pour élargir l'exploration de l'espace de recherche et éviter la convergence prématurée qui peut être causée par la classe A.

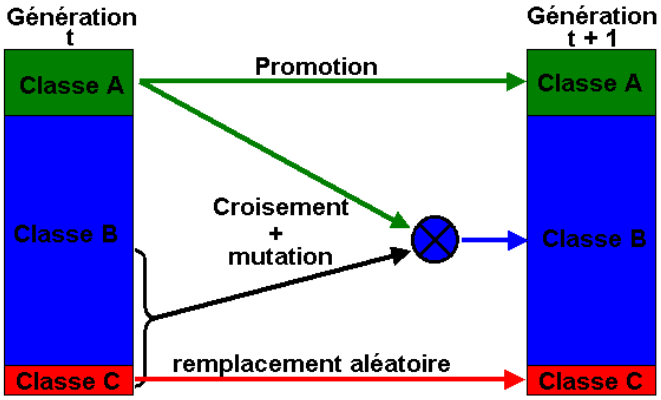



Figure 5.6 : Formation de la génération suivante dans GAOSPF.

### Les opérateurs génétiques

La procédure de croisement utilisée est appelée ''clés aléatoires '',elles est inspirer du croisement uniforme du codage binaire. Pour combiner deux solutions parentes $p_1 \in A(elite)$ et $p_2 \in (B \cup C)(nonelite)$ , on génère aléatoirement un vecteur r à $\mid A \mid$ composantes réelles dans l'intervalle [0,1]. le nombre réel$p_c \in [0.5, 1]$ permet de déterminer si un gène est hérité de $p_1$ ou $p_2$ . un enfant e est alors générer par l'application de l'algorithme de la figure

Pour éviter la convergence vers un optimum local et diversifier la recherche, l'opérateur de mutation à été inséré dans l'algorithme précédent : on génère un vecteur s à $\mid A \mid$ composantes réelles dans l'intervalle [0,1], un enfant e est alors construit comme il est indiqué dans l'algorithme

de la figure 11 , où $p_m$ désigne la probabilité de mutation.

**Algorithm 4** L'opération de croisement

Pour tout gènes $i = 1....$ | $A$ |
Faire
Si $r[i] < p_c$ Alors $e[i] := p_1[i]$
Sinon $e[i] := p_2[i]$

**Algorithm 5** L'algorithme de production d'un enfant dans GAOSPF

Fonction Crois-mutate($p_1, p_2 : parents$);
$Retournee : enfant$;
$Pourtoutgnes$i=1....| $A$ |
Faire
Si $s[i] < p_m$ Alors $e[i] := random[1, w_{max}]$
Sinon Si $r[i] < p_c$ Alors $e[i] := p_1[i]$
Sinon $e[i] := p_2[i]$

| | | | | | | |
|---|---|---|---|---|---|---|
| $P_m$ | 0.720 | 0.310 | 0.008 | 0.052 | 0.011 | 0.027 |
| $P_c$ | 0.300 | 0.729 | 0.531 | 0.215 | 0.823 | 0.026 |
| $P_1$ | 6 | 14 | 9 | 8 | 6 | 7 |
| $P_2$ | 10 | 8 | 1 | 15 | 3 | 10 |
| E | 6 | 8 | 12 | 15 | 3 | 7 |

Figure 5.7 :Exemple d'application de la fonction Crois-mutate..

Avec cette stratégie,on a pu constater qu'une meilleure convergence est obtenue avec une probabilité de croisement $p_c = 0.7$ et une probabilité de mutation $p_m = 0.01$ . En général, cet algorithme appliqué sur divers exemples, a donné de bonnes solutions pour une taille de population npop = 200 après maxgen = 700 générations.

# Chapitre 6

# Résultats

Notre but est de trouver un meilleur réglage des poids pour le routage OSPF par un algorithme génétique efficace.Dans ce dernier chapitre, nous commençons par définir les valeurs de tous les paramètres utilisés dans l'algorithme, ensuitenous décrivons le jeu de tests sur lequel notre algorithme a été appliqué, nous terminons ce chapitre par une étude comparative des résultats obtenus avec celles mentionnées dans fortz et al[ 26].

## 6.1 Technologies utilisées

### Java

Le langage Java est un langage généraliste de programmation synthétisant les principaux langages existants lors de sa création en 1995 par Sun Microsystems. Il permet une programmation orientée-objet (à l'instar de SmallTalk et, dans une moindre mesure, C++), modulaire (langage ADA) et reprend une syntaxe très proche de celle du langage C. Outre son orientation objet, le langage Java a l'avantage d'être modulaire (on peut écrire des portions de code génériques, c-à-d utilisables par plusieurs applications), rigoureux (la plupart des erreurs se produisent à la compilation et non à l'exécution) et portable (un même programme compilé peut s'exécuter sur différents environnements). En contre-partie, les applications Java ont le défaut d'être plus lentes à l'exécution que des applications programmées en C par exemple.


FIGURE 6.1 – Logo de Java.

## JavaFx

JavaFX est une technologie créée par Sun Microsystems qui appartient désormais à Oracle. Avec l'apparition de Java 8 en mars 2014, JavaFX devient la bibliothèque de création d'interface graphique officielle du langage Java, pour toutes les sortes d'application (applications mobiles, applications sur poste de travail, applications Web), le développement de son prédécesseur Swing étant abandonné (sauf pour les corrections de bogues). JavaFX contient des outils très divers, notamment pour les médias audio et vidéo, le graphisme 2D et 3D, la programmation Web, la programmation multi-fils etc. Le SDK de JavaFX étant désormais intégré au JDK standard Java SE


FIGURE 6.2 – Logo de JavaFx.

## Scene Builder

Scene Builder est un outil interactif de conception d'interface graphique pour JavaFX. Créé par Oracle, il permet de construire rapidement des interfaces utilisateurs sans avoir besoin de (savoir) coder. Le logiciel a été publié pour la première fois le 14 août 2012 par Oracle sous le nom JavaFX Scene Builder

FIGURE 6.3 – Logo de Scene Builder.

## Eclipse

Eclipse IDE est un environnement de développement intégré libre (le terme Eclipse désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions


FIGURE 6.4 – Logo de Eclipse.

## 6.2 Conditions expérimentales

Pour tous les tests, nous considérons une population de taille npop = 200, le nombre de générations maximal est arbitrairement fixé à maxgen = 300. Les opérateurs génétiques utilisés sont ceux décrits dans la section 5.4. Nous avons constaté expérimentalement que les réglages des paramètres de l'algorithme génétique GAOSPF proposés dans Ericsson [19] sont optimaux :

• Le paramètre $\alpha = 0.2$ : la proportion des élites dans la population constituant la super-classe A ;

• Le paramètre $\beta = 0.1$ : la proportion des plus mauvais individus de la population constituant la classe C ;

• La probabilité de croisement $P_c = 0.7$ ;

• La probabilité de mutation $P_m = 0.01$ ;

• La borne supérieure pour les valeurs des poids $W_{max} = 20$ .

## 6.3 Présentation du jeu de données

Afin de valider et tester numériquement l'algorithme proposé, nous avons utilisé les mêmes réseaux expérimentaux et les mêmes matrices de demandes utilisés dans Ericsson[19]. Il s'agit de 3 réseaux à 2-niveaux, 2 réseaux Waxman et 2 réseaux pure Random figure 6.2

| Type de réseau | Nombre de nœuds | Nombre d'arcs |
|---|---|---|
| Graphe Hierarchique 2-niveaux | 50 | 148 |
| Graphe Hierarchique 2-niveaux | 50 | 212 |
| Graphe Hierarchique 2-niveaux | 100 | 280 |
| Graphe pure Random | 50 | 245 |
| Graphe pure Random | 50 | 228 |
| Graphe Waxman | 50 | 169 |
| Graphe Waxman | 50 | 230 |

Les Réseaux utilisés dans l'expérimentation.

## 6.4 Les heuristiques

Nous comparons nos résultats avec ceux obtenus, sur les mêmes instances, par plusieurs heuristiques et en particulier la référence LPLB :

**InvCapOSPF :**

recommander par CISCO, le poids d'un arc est inversement proportionnel à sa capacité (chapitre 3)

**UnitOSPF :**

tous les poids d'arcs ont la valeur 1, ainsi le chemin est calculé en nombre de sauts . C'est la première métrique dans l'histoire du routage.

**RandomOSPF :**

cette heuristique assigne à chaque arc un poids choisi aléatoirement entre certaines bornes

**LPLB (Linear Programming Lower Bound) :**

CPLEX

**FT :**

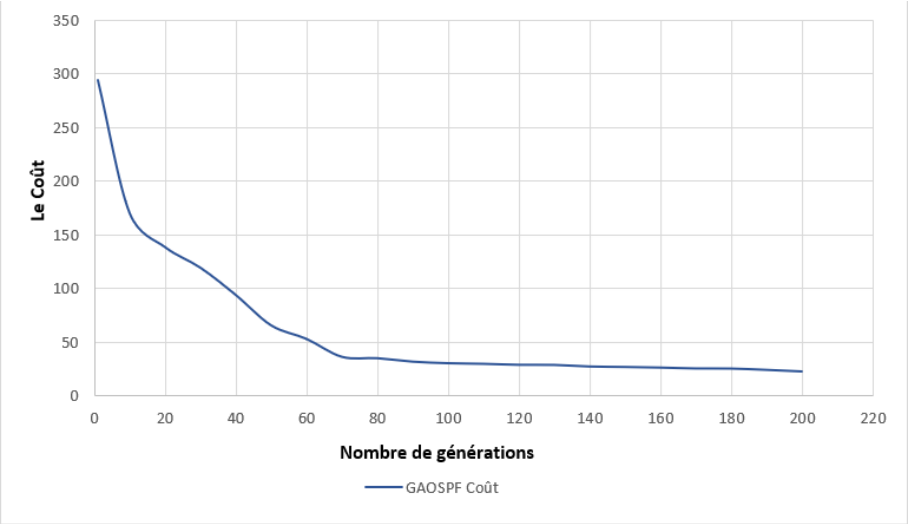
Les poids d'arcs sont obtenus par l'application de l'algorithme Tabou proposé par Fortz et Thorup

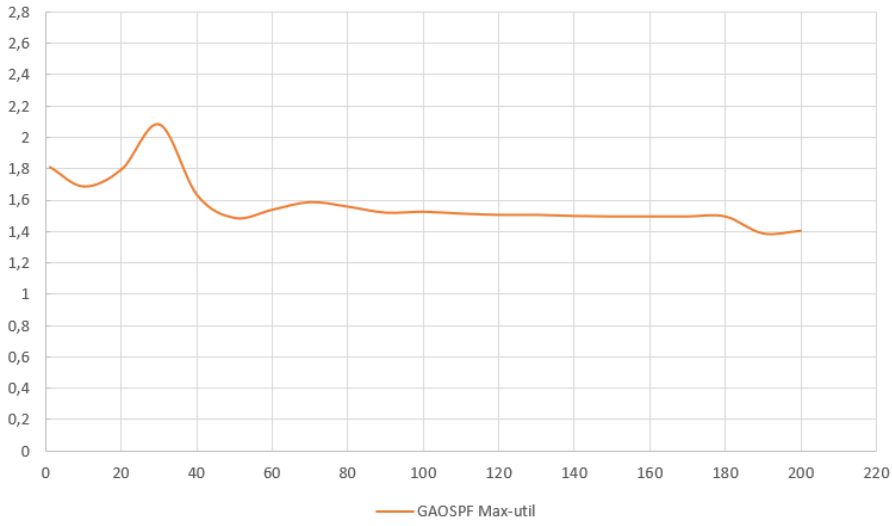
## 6.5 Les résultats obtenues

Nous observons que l'algorithme génétique GAOSPF trouve généralement de bonnes solutions proches de la borne inférieure. À l'exception des réseaux Random, GAOSPF a trouvé des solutions de qualité similaire à celles trouvées par la recherche Taboue de Fortz et Thorup [26]. Les solutions trouvées par l'algorithme GAOSPF est significativement meilleures que celles produites par UnitOSPF, InvCapOSPF et RandomOSPF, et permettre une augmentation significative du débit du réseau par rapport aux débits associés à ces heuristiques.

| Nombre de générations | GAOSPF | |
|---|---|---|
| | Coût | Max-util |
| 1 | 293,76 | 1,812 |
| 10 | 169,805 | 1,689 |
| 20 | 138,258 | 1,798 |
| 30 | 119,08 | 2,083 |
| 40 | 93,753 | 1,634 |
| 50 | 65,452 | 1,489 |
| 60 | 52,85 | 1,542 |
| 70 | 36,192 | 1,59 |
| 80 | 34,925 | 1,561 |
| 90 | 31,701 | 1,523 |
| 100 | 30,259 | 1,529 |
| 110 | 29,689 | 1,517 |
| 120 | 28,776 | 1,509 |
| 130 | 28,67 | 1,509 |
| 140 | 27,196 | 1,502 |
| 150 | 26,743 | 1,499 |
| 160 | 26,201 | 1,499 |
| 170 | 25,385 | 1,499 |
| 180 | 25,286 | 1,499 |
| 190 | 24,074 | 1,391 |
| 200 | 22,596 | 1,408 |

FIGURE 6.5 – L'évolution d'une population de 200 individus jusqu'à la génération 200




## 6.6 Les résultats

Nous clôturons ce chapitre en présentant les résultats expérimentaux obtenus sur un nombre assez important d'instances (figure ). Nous comparons nos résultats avec ceux des six heuristiques décrites dans la section précédente. Pour chaque instance, nous montrons sur un tableau suivi de deux graphes les résultats obtenus. Dans chaque tableau on présente le coût du routage ainsi que l'utilisation maximale (donnée entre parenthèses) en fonction du cumul des demandes. Notons que le coût est normalisé par le facteur de normalisation $\Phi_{uncap}$ , comme il a été mentionné dans le chapitre précédent. Sur le graphe montrant le coût du routage en fonction des demandes, la ligne horizontale représente le seuil 10.667 qui indique que la congestion du réseau est atteinte.Sur le graphe représentant l'utilisation maximale en fonction du cumul des demandes, la ligne horizontale représente l'égalité de la charge et de la capacité $(l(a) = C(a))$ .

On constate que les résultats obtenus par notre algorithme sont meilleurs globalement que ceux obtenus par les autres heuristiques et s'approchent le plus de la référence LPLB.

| D | UnitOSPF | InvCap | RandomOSPF | F&T | GAOSPF | LPLB |
|---|---|---|---|---|---|---|
| 411 | 1 (0,19) | 1,02 (0,22) | 1,03 (0,18) | 1 (0,17) | 1 (0,17) | 1 (0,90) |
| 812 | 1 (0,37) | 1,03 (0,43) | 1,03 (0,39) | 1 (0,33) | 1 (0,31) | 1 (0,19) |
| 1232 | 1,04 (0,56) | 1,06 (0,65) | 1,08 (0,55) | 1,01 (0,33) | 1,01 (0,33) | 1,01 (0,28) |
| 1643 | 1,13 (0,75) | 1,15 (0,87) | 1,17 (0,58) | 1,05 (0,55) | 1,05 (0,56) | 1,04 (0,38) |
| 2053 | 1,31 (0,93) | 2,34 (1,08) | 1,21 (0,80) | 1,11 (0,67) | 1,11 (0,67) | 1,1 (0,47) |
| 2464 | 4,67 (1,12) | 21,89 (1,30) | 1,5 (0,85) | 1,2 (0,66) | 1,2 (0,67) | 1,17 (0,56) |
| 2874 | 21,32 (1,31) | 37,73 (1,51) | 2,63 (1,03) | 1,31 (0,79) | 1,34 (0,86) | 1,27 (0,66) |
| 3285 | 50,3 (1,5) | 56,18 (1,73) | 9,88 (1,15) | 1,44 (0,90) | 1,47 (0,89) | 1,39 (0,75) |
| 3696 | 79,26 (1,68) | 75,97 (1,95) | 36,96 (1,45) | 1,66 (0,90) | 1,69 (0,95) | 1,53 (0,85) |
| 4106 | 119,52 (1,87) | 106,9 (2,16) | 55,77 (1,48) | 2,2 (1,00) | 2,23 (1,00) | 1,89 (0,94) |
| 4517 | 168,67 (2,06) | 140,52 (2,38) | 157,26 (1,57) | 4,29 (1,10) | 5,11 (1,10) | 3,44 (1,03) |
| 4928 | 222,01 (2,24) | 180,3 (2,60) | 223,01 (1,74) | 15,73 (1,21) | 21,99 (1,34) | 14,4 (1,13) |

FIGURE 6.6 – Le coût du routage et (l'utilisation maximale) pour le réseau hiérarchique 2-niveaux (50 noeuds et 148 arcs) avec différentes demandes.

| D | UnitOSPF | InvCap | RandomOSPF | F&T | GAOSPF | LPLB |
|---|---|---|---|---|---|---|
| 280 | 1 (0,19) | 1 (0,19) | 1,05 (0,20) | 1 (0,18) | 1 (0,19) | 1 (0,07) |
| 560 | 1,01 (0,39) | 1,01 (0,38) | 1,06 (0,32) | 1 (0,33) | 1 (0,33) | 1 (0,15) |
| 841 | 1,04 (0,58) | 1,04 (0,56) | 1,11 (0,60) | 1,02 (0,33) | 1,02 (0,33) | 1,01 (0,22) |
| 1121 | 1,11 (0,78) | 1,11 (0,75) | 1,19 (0,67) | 1,06 (0,43) | 1,06 (0,43) | 1,03 (0,30) |
| 1401 | 1,34 (0,97) | 1,27 (0,94) | 1,29 (0,72) | 1,09 (0,55) | 1,09 (0,55) | 1,06 (0,37) |
| 1681 | 12,28 (1,17) | 6,28 (1,13) | 1,48 (0,83) | 1,14 (0,67) | 1,15 (0,67) | 1,11 (0,45) |
| 1962 | 33,71 (1,36) | 27,66 (1,31) | 1,98 (0,99) | 1,21 (0,72) | 1,23 (0,77) | 1,16 (0,52) |
| 2242 | 50,66 (1,56) | 44,14 (1,50) | 3,24 (1,04) | 1,32 (0,83) | 1,33 (0,85) | 1,24 (0,60) |
| 2522 | 73,84 (1,75) | 63,9 (1,69) | 13,98 (1,20) | 1,46 (0,90) | 1,46 (0,90) | 1,35 (0,67) |
| 2802 | 102,83 (1,95) | 95,13 (1,88) | 41,77 (1,32) | 1,73 (0,95) | 1,79 (1,00) | 1,47 (0,75) |
| 3082 | 135,13 (2,14) | 128,35 (2,06) | 100,3 (2,33) | 2,2 (1,00) | 2,23 (1,00) | 1,61 (0,82) |
| 3363 | 167,47 (2,34) | 159,85 (2,25) | 141,12 (1,70) | 4,52 (1,10) | 5,09 (1,10) | 1,83 (0,89) |

FIGURE 6.7 – Le coût du routage et (l'utilisation maximale) pour le réseau hiérarchique 2-niveaux (50 noeuds et 212 arcs) avec différentes demandes

| D | UnitOSPF | InvCap | F&T | GAOSPF | LPLB |
|---|---|---|---|---|---|
| 384 | 1 (0,18) | 1,02 (0,17) | 1 (0,17) | 1 (0,18) | 1 (0,11) |
| 768 | 1 (0,36) | 1,02 (0,35) | 1 (0,33) | 1 (0,33) | 1 (0,22) |
| 1151 | 1,02 (0,54) | 1,03 (0,52) | 1,01 (0,36) | 1,01 (0,36) | 1,01 (0,34) |
| 1535 | 1,1 (0,72) | 1,09 (0,69) | 1,03 (0,51) | 1,03 (0,52) | 1,03 (0,45) |
| 1919 | 1,17 (0,90) | 1,17 (0,86) | 1,08 (0,63) | 1,08 (0,63) | 1,06 (0,56) |
| 2303 | 1,89 (1,07) | 1,59 (1,04) | 1,13 (0,67) | 1,14 (0,67) | 1,11 (0,67) |
| 2686 | 11,65 (1,25) | 8,87 (1,21) | 1,22 (0,78) | 1,23 (0,78) | 1,2 (0,78) |
| 3070 | 21,04 (1,43) | 17,5 (1,38) | 1,31 (0,89) | 1,34 (0,89) | 1,28 (0,89) |
| 3454 | 37,93 (1,61) | 24,93 (1,56) | 1,55 (1,01) | 1,63 (1,01) | 1,52 (1,01) |
| 3838 | 67,6 (1,79) | 38,54 (1,73) | 3,25 (1,12) | 3,27 (1,12) | 3,18 (1,12) |
| 4221 | 117,09 (1,97) | 70,25 (1,90) | 11,83 (1,23) | 12,17 (1,23) | 11,71 (1,23) |
| 4605 | 168,47 (2,15) | 114,55 (2,07) | 19,26 (1,34) | 19,66 (1,34) | 19,06 (1,34) |

FIGURE 6.8 – Le coût du routage et (l'utilisation maximale) pour le réseau hiérarchique 2-niveaux (100 noeuds et 280 arcs) avec différentes demandes

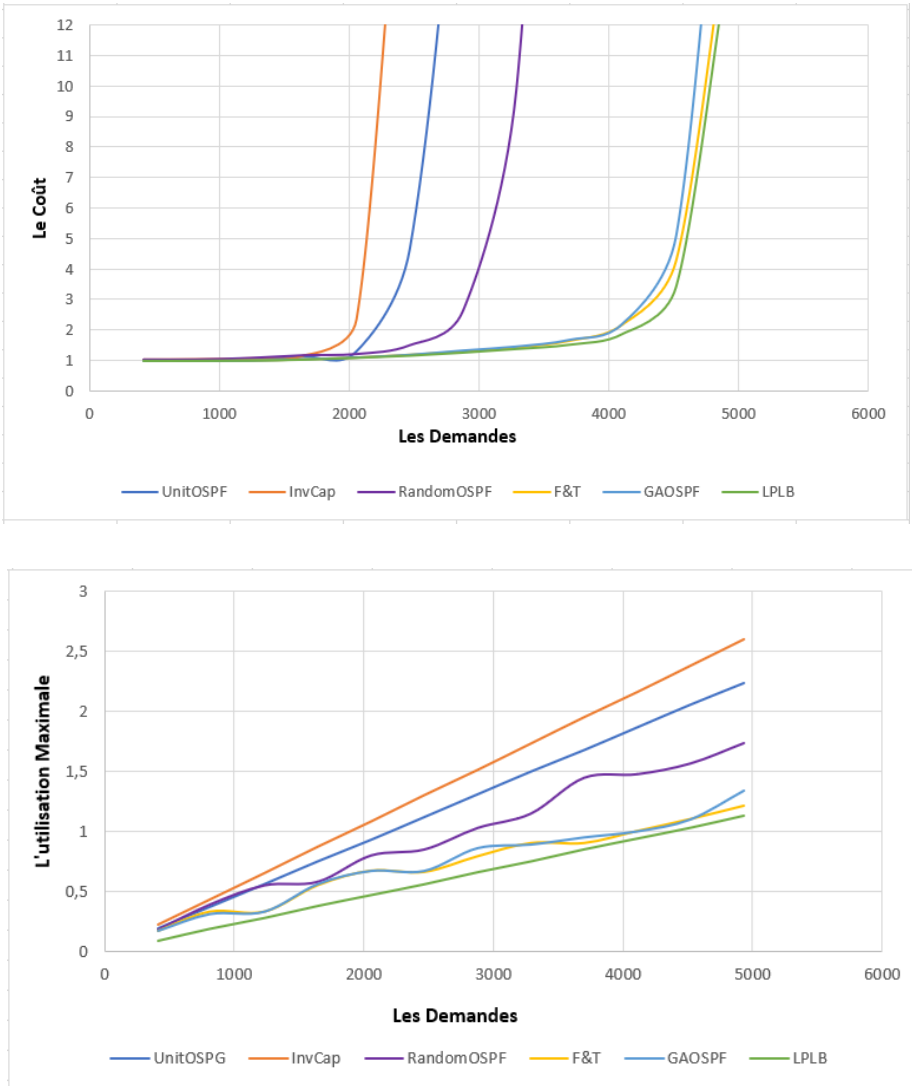
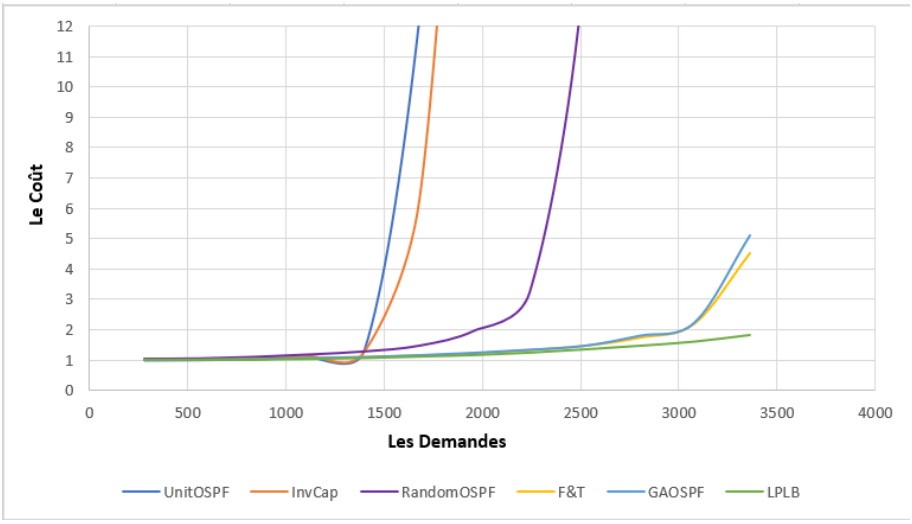



FIGURE 6.9 – Le coût du routage et l'utilisation maximale en fonction des demandes sur le réseau hiérarchique 2-niveaux (50 noeuds et 148 arcs)



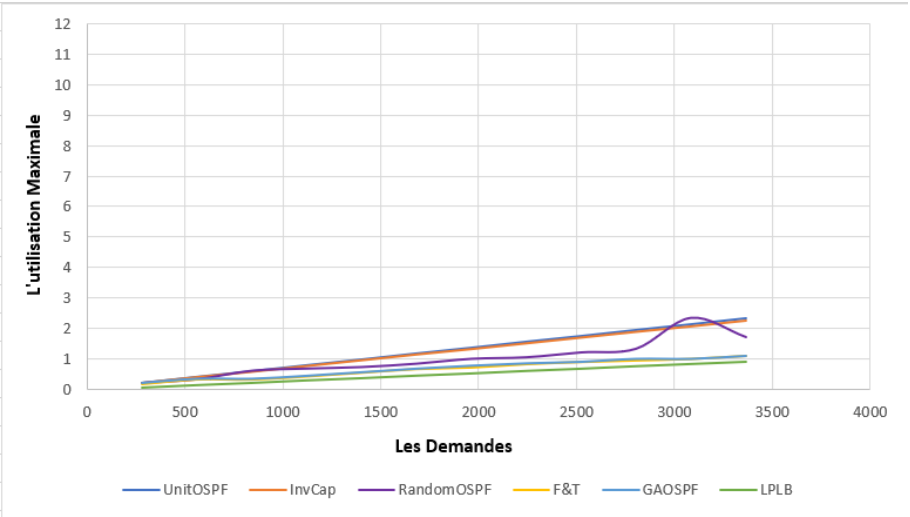
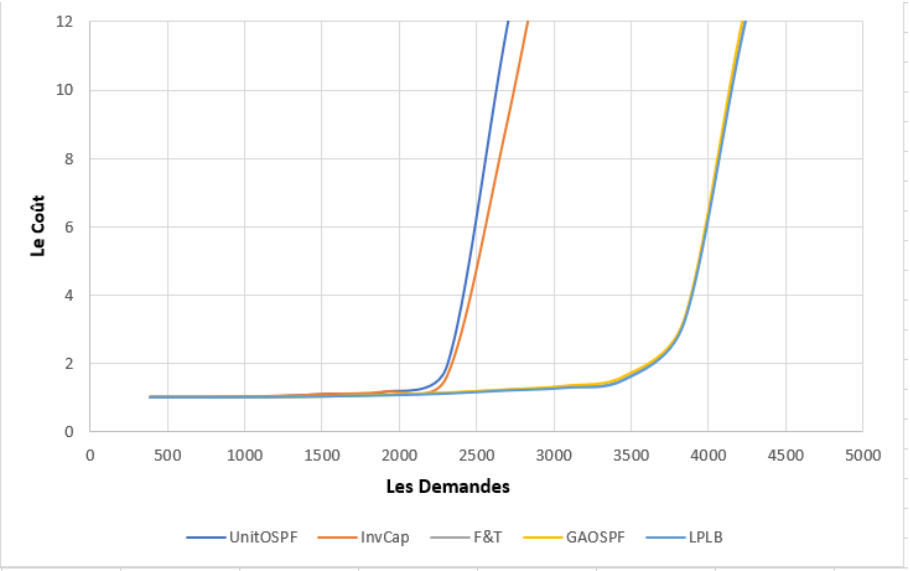



FIGURE 6.10 – Le coût du routage et l'utilisation maximale en fonction des demandes sur le réseau hiérarchique 2-niveaux (50 noeuds et 212 arcs)..



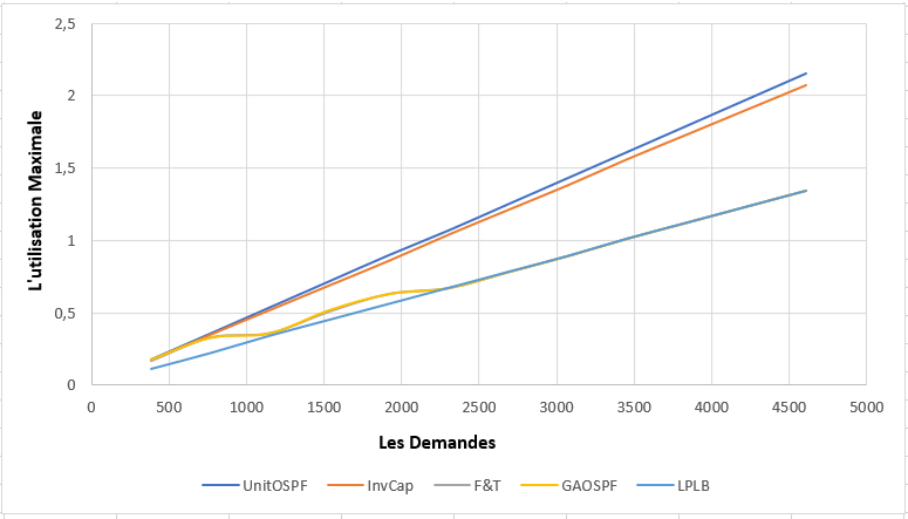



FIGURE 6.11 – Le coût du routage et l'utilisation maximale en fonction des demandes sur le réseau hiérarchique 2-niveaux (100 noeuds et 280 arcs)..

| InvCap | RandomOSPF | F&T | GaOSPF | LPLB |
|---|---|---|---|---|
| 1 (0,16) | 1,12 (0,29) | 1 (0,22) | 1 (0,20) | 1 (0,10) |
| 1 (0,32) | 1,16 (0,43) | 1 (0,31) | 1 (0,32) | 1 (0,20) |
| 1,04 (0,47) | 1,35 (0,75) | 1 (0,33) | 1,02 (0,35) | 1 (0,30) |
| 1,14 (0,63) | 1,74 (0,89) | 1,04 (0,47) | 1,1 (0,59) | 1,03 (0,40) |
| 1,3 (0,79) | 3,49 (1,04) | 1,15 (0,63) | 1,24 (0,67) | 1,13 (0,50) |
| 1,57 (0,95) | 39,47 (1,29) | 1,29 (0,67) | 1,41 (0,73) | 1,27 (0,61) |
| 3,65 (1,10) | 126,24 (2,05) | 1,46 (0,71) | 1,65 (0,89) | 1,42 (0,71) |
| 27,35 (1,26) | 222,39 (1,97) | 1,69 (0,87) | 1,98 (0,90) | 1,61 (0,81) |
| 66,67 (1,42) | 422,8 (2,55) | 1,99 (0,91) | 2,77 (1,00) | 1,9 (0,91) |
| 122,87 (1,58) | 514,03 (2,90) | 2,65 (1,01) | 5,19 (1,09) | 2,43 (1,01) |
| 188,78 (1,73) | 667,62 (2,54) | 5,22 (1,11) | 25,36 (1,21) | 4,26 (1,11) |
| 264,61 (1,89) | 849,1 (2,99) | 17,04 (1,21) | 84,56 (1,79) | 13,75 (1,21) |

FIGURE 6.12 – Le coût du routage et (l'utilisation maximale) pour le réseau pure Random (50 noeuds et 228 arcs) avec différentes demandes.

| D | UnitOSPF | InvCap | RandomOSPF | F&t | GAOSPF | LPLB |
|---|---|---|---|---|---|---|
| 4463 | 1 (0,21) | 1 (0,21) | 1,13 (0,35) | 1 (0,25) | 1 (0,21) | 1 (0,10) |
| 8927 | 1,01 (0,43) | 1,01 (0,43) | 1,24 (0,60) | 1 (0,32) | 1 (0,34) | 1 (0,20) |
| 13390 | 1,07 (0,64) | 1,07 (0,64) | 1,58 (0,87) | 1,01 (0,35) | 1,05 (0,49) | 1,01 (0,30) |
| 17854 | 1,22 (0,86) | 1,22 (0,86) | 5,37 (1,09) | 1,08 (0,67) | 1,16 (0,66) | 1,05 (0,41) |
| 22317 | 2,13 (1,07) | 2,13 (1,07) | 20,24 (1,17) | 1,22 (0,67) | 1,36 (0,80) | 1,19 (0,51) |
| 26781 | 16,6 (1,28) | 16,6 (1,28) | 165,63 (1,68) | 1,4 (0,67) | 1,62 (0,90) | 1,35 (0,61) |
| 31244 | 42,65 (1,50) | 42,65 (1,50) | 335,25 (2,35) | 1,63 (0,89) | 2,1 (0,99) | 1,57 (0,71) |
| 35708 | 81,28 (1,71) | 81,28 (1,71) | 558,74 (2,17) | 2,01 (0,90) | 3,72 (1,08) | 1,87 (0,81) |
| 40171 | 131,86 (1,92) | 131,86 (1,92) | 615,9 (3,58) | 2,65 (0,99) | 12,69 (1,19) | 2,29 (0,91) |
| 44635 | 203,56 (2,14) | 203,56 (2,14) | 890,17 (3,04) | 5,11 (1,09) | 34,32 (1,49) | 3,02 (1,01) |
| 49098 | 279 (2,35) | 279 (2,35) | 1067,68 (3,01) | 14,09 (1,11) | 114,52 (1,77) | 5,98 (1,11) |
| 53562 | 357,04 (2,57) | 357,04 (2,57) | 1214,61 (3,59) | 52,68 (1,57) | 206,79 (2,10) | 19,65 (1,22) |

FIGURE 6.13 – Le coût du routage et l'utilisation maximale en fonction des demandes sur le réseau hiérarchique 2-niveaux (50 noeuds et 245 arcs).

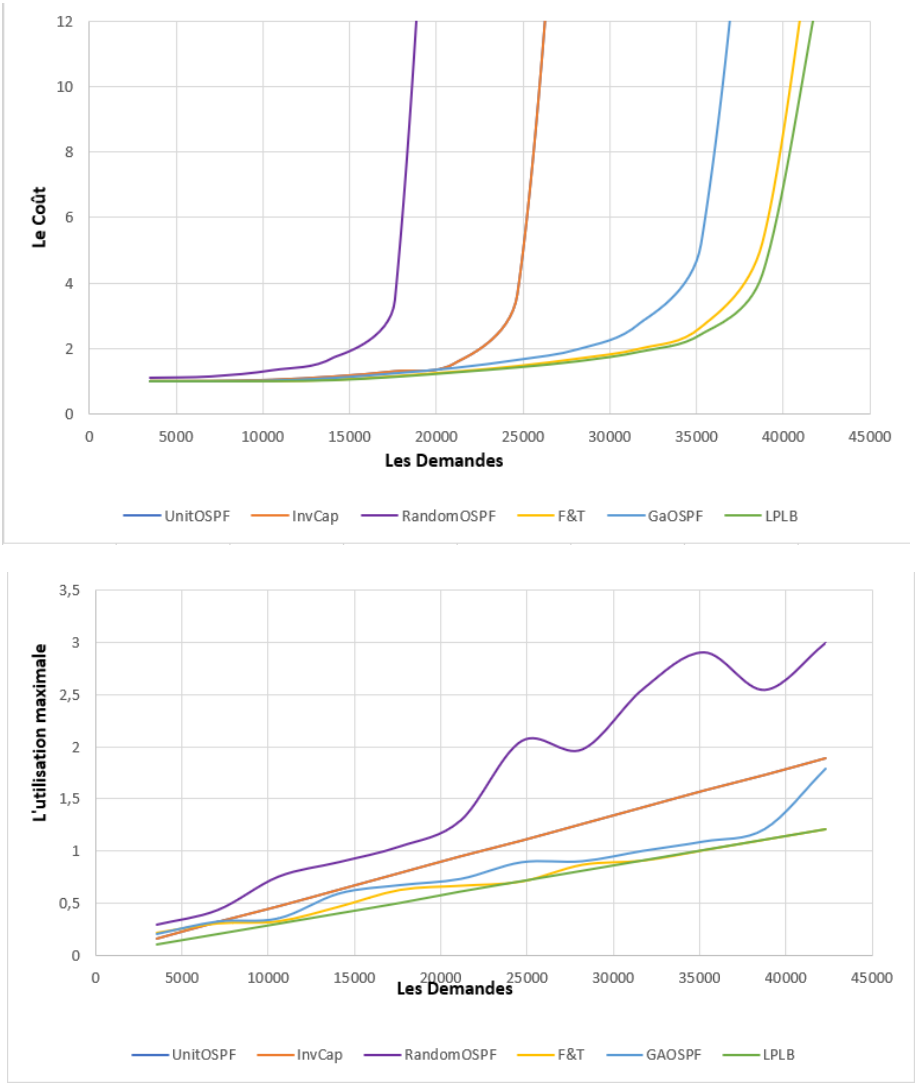
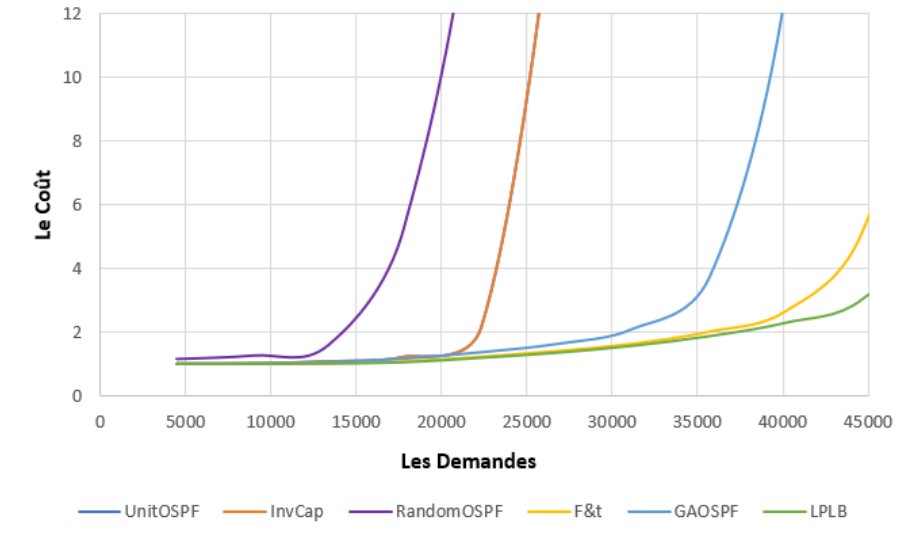



FIGURE 6.14 – Le coût du routage et l'utilisation maximale en fonction des demandes sur le réseau hiérarchique 2-niveaux (50 noeuds et 148 arcs).



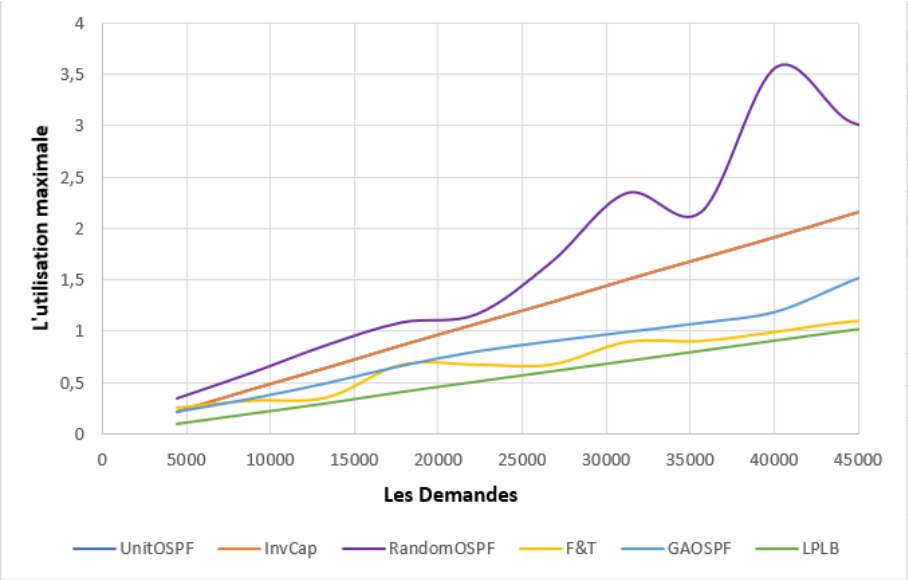


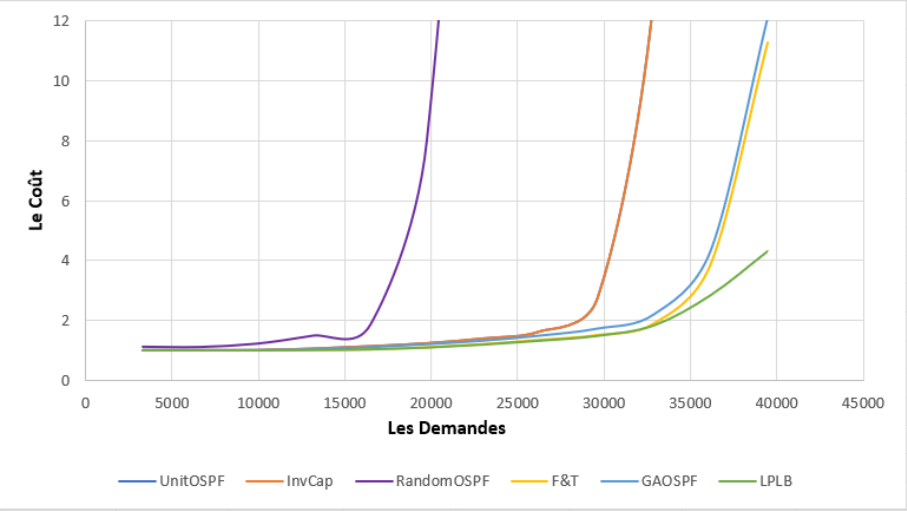

FIGURE 6.15 – Le coût du routage et l'utilisation maximale en fonction des demandes sur le réseau hiérarchique 2-niveaux (50 noeuds et 148 arcs)..

| D | UnitOSPF | InvCap | RandomOSPF | F&T | GAOSPF | LPLB |
|---|---|---|---|---|---|---|
| 3287 | 1 (0,12) | 1 (0,12) | 1,13 (0,20) | 1 (0,13) | 1 (0,12) | 1 (0,10) |
| 6574 | 1 (0,23) | 1 (0,23) | 1,12 (0,38) | 1 (0,27) | 1 (0,23) | 1 (0,20) |
| 9862 | 1 (0,35) | 1 (0,35) | 1,23 (0,63) | 1 (0,33) | 1 (0,34) | 1 (0,31) |
| 13149 | 1,05 (0,46) | 1,05 (0,46) | 1,5 (0,70) | 1,01 (0,41) | 1,03 (0,45) | 1,01 (0,41) |
| 16436 | 1,13 (0,58) | 1,13 (0,58) | 1,84 (0,94) | 1,03 (0,51) | 1,1 (0,53) | 1,03 (0,51) |
| 19723 | 1,23 (0,70) | 1,23 (0,70) | 7,99 (1,15) | 1,1 (0,65) | 1,2 (0,62) | 1,09 (0,61) |
| 23011 | 1,39 (0,81) | 1,39 (0,81) | 37,04 (1,35) | 1,21 (0,71) | 1,32 (0,73) | 1,19 (0,71) |
| 26298 | 1,63 (0,93) | 1,63 (0,93) | 104,8 (1,54) | 1,35 (0,81) | 1,49 (0,81) | 1,32 (0,81) |
| 29585 | 2,71 (1,04) | 2,71 (1,04) | 248,27 (1,86) | 1,51 (0,92) | 1,72 (0,92) | 1,48 (0,92) |
| 32872 | 12,82 (1,16) | 12,82 (1,16) | 272,03 (2,17) | 1,9 (1,02) | 2,2 (1,02) | 1,83 (1,02) |
| 36159 | 38,71 (1,28) | 38,71 (1,28) | 515,65 (2,06) | 3,89 (1,12) | 4,34 (1,12) | 2,84 (1,12) |
| 39447 | 78,08 (1,39) | 78,08 (1,39) | 587,54 (2,50) | 11,29 (1,22) | 12,11 (1,22) | 4,3 (1,22) |

FIGURE 6.16 – Le coût du routage et (l'utilisation maximale) pour le réseau Waxman (50 noeuds et 169 arcs) avec différentes demandes..

| D | UnitOSPF | InvCap | RandomOSPF | F&T | GAOSPF | LPLB |
|---|---|---|---|---|---|---|
| 2118 | 1 (0,14) | 1 (0,14) | 1,05 (0,18) | 1 (0,15) | 1 (0,15) | 1 (0,10) |
| 4235 | 1 (0,28) | 1 (0,28) | 1,07 (0,33) | 1 (0,23) | 1 (0,28) | 1 (0,21) |
| 6353 | 1,01 (0,42) | 1,01 (0,42) | 1,11 (0,45) | 1 (0,33) | 1 (0,33) | 1 (0,31) |
| 8470 | 1,08 (0,55) | 1,08 (0,55) | 1,24 (0,63) | 1,02 (0,42) | 1,02 (0,42) | 1,02 (0,42) |
| 10588 | 1,17 (0,69) | 1,17 (0,69) | 1,42 (0,80) | 1,09 (0,58) | 1,14 (0,62) | 1,08 (0,52) |
| 12706 | 1,32 (0,83) | 1,32 (0,83) | 1,69 (0,87) | 1,18 (0,66) | 1,19 (0,67) | 1,18 (0,62) |
| 14823 | 1,6 (0,97) | 1,6 (0,97) | 2,42 (0,97) | 1,31 (0,73) | 1,32 (0,73) | 1,29 (0,73) |
| 16941 | 3,9 (1,11) | 3,9 (1,11) | 6,66 (1,10) | 1,48 (0,84) | 1,5 (0,84) | 1,45 (0,83) |
| 19059 | 20,63 (1,25) | 20,63 (1,25) | 45,92 (1,31) | 1,76 (0,94) | 1,78 (0,94) | 1,72 (0,94) |
| 21176 | 45,77 (1,39) | 45,77 (1,39) | 98,08 (1,37) | 2,37 (1,04) | 2,4 (1,04) | 2,31 (1,04) |
| 23294 | 84,41 (1,52) | 84,41 (1,52) | 198,25 (1,57) | 6,02 (1,14) | 6,18 (1,25) | 3,27 (1,14) |
| 25411 | 139,52 (1,66) | 139,52 (1,66) | 314,38 (2,09) | 13,15 (1,25) | 13,79 (1,25) | 4,36 (1,25) |

FIGURE 6.17 – Le coût du routage et (l'utilisation maximale) pour le réseau Waxman (50 noeuds et 230 arcs) avec différentes demandes.



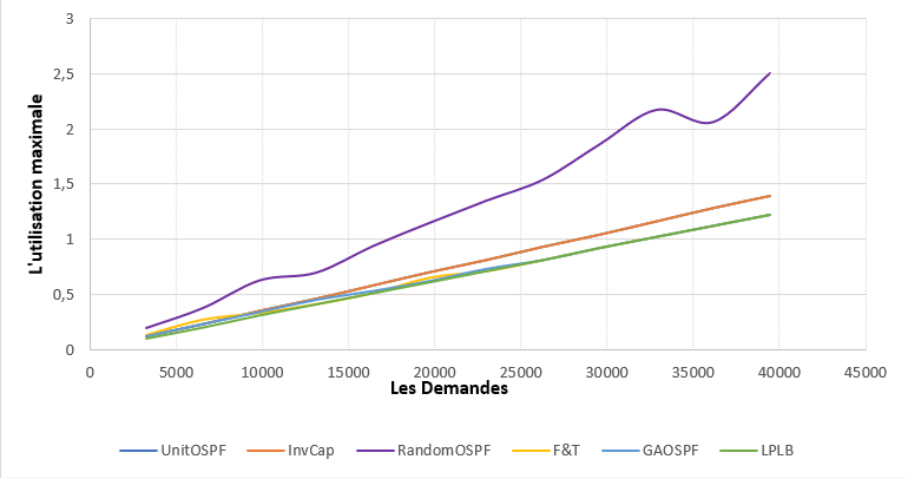
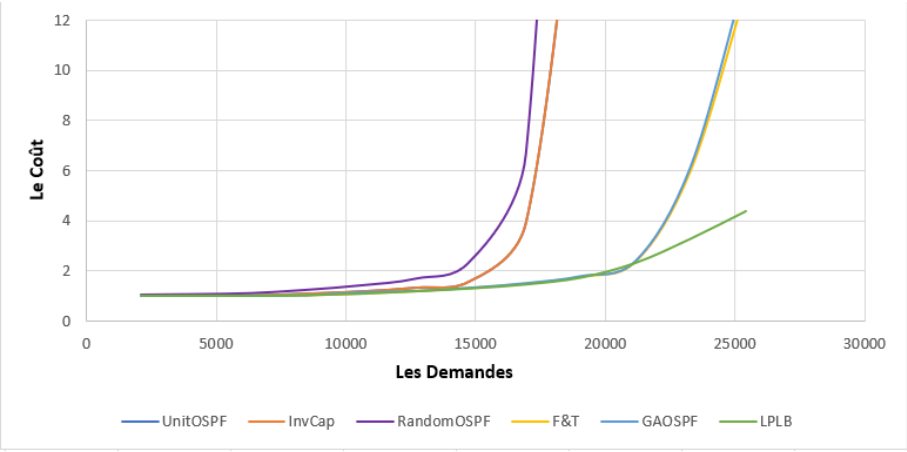




FIGURE 6.18 – Le coût du routage et (l'utilisation maximale) pour le réseau Waxman (50 noeuds et 169 arcs) avec différentes demandes..



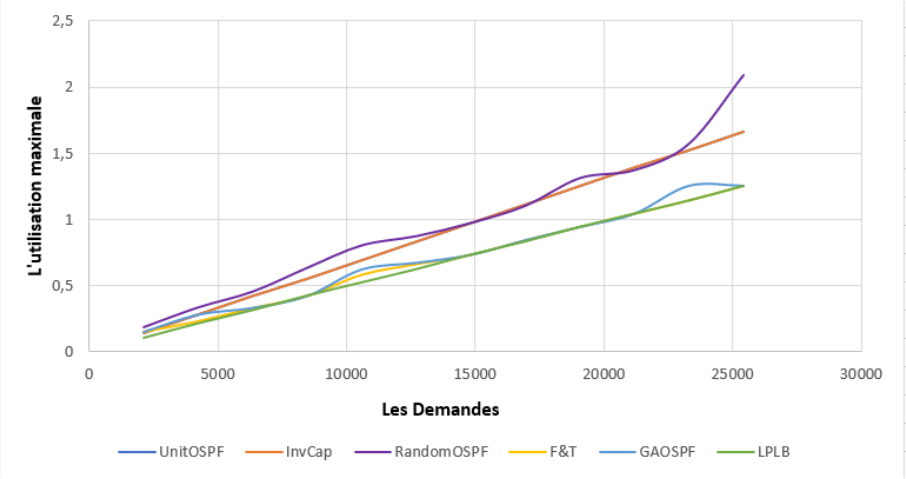



FIGURE 6.19 – Le coût du routage et (l'utilisation maximale) pour le réseau Waxman (50 noeuds et 230 arcs) avec différentes demandes..

# Conclusion générale

Aujourd'hui, la grande performance des métaheuristiques commencent à être reconnue. De très nombreuses méthodes évolutionnaires avaient été expérimentées, très souvent avec succès, pour résoudre divers problèmes d'optimisation combinatoire. Dans ce contexte, nous nous sommes fixés comme objectif la conception d'une méthode évolutionnaire pour résoudre le problème de réglage des poids du routage OSPF, il s'agit de l'algorithme proposé par M. Ericsson et al. [9]. Après avoir modélisé la forme de l'algorithme génétique utilisé, nous l'avons implémenté en Java, puis nous avons testé sa validité sur un jeu de données s'approchant le plus de la réalité du trafic sur un système autonome du réseau Internet, pour lesquelles des résultats encourageants ont été obtenus, tant pour la rapidité que pour la précision.

Il serait toujours possible d'améliorer l'algorithme proposé dans ce mémoire et de le mettre à jour grâce à des modules implémentant les nouvelles techniques à venir. En effet, la recherche sur les algorithmes génétiques a sans doute encore de beaux jours devant elle, étant donné qu'ils constituent souvent l'une des rare méthodes envisageables pour obtenir des résultats sur des problèmes où les méthodes classiques échouent. Pour avoir des résultats plus précis, il serait souhaitable de faire suivre notre algorithme par une méthode de recherche locale ou d'une recherche taboue comme celle proposée par B. Fortz et M. Thorup [26].

Une dernière perspective, concerne la prise en compte de la qualité de service (Qos) dans l'Internet. La QoS se décline principalement en quatre paramètres : débit, latence, gigue et perte. Le débit ( ou bande passante ) représente les ressources de transmission occupées par un flot . La latence correspond au délai de transfert de bout en bout d'un paquet. La gigue correspond aux variations de latence des paquets. La gigue provient essentiellement des variations de trafic sur le lien de sortie des routeurs. La perte de paquets se produit principalement lorsque l'intensité du trafic sur le lien de sortie devient supérieure à leur capacité d'écoulement. Il s'agit alors d'une indication de congestion. On peut donc utiliser, en plus de la matrice des demandes, une matrice de délai pour satisfaire le critère QoS.

L'idée qui paraît la plus simple est de pondérer la matrice de délai après normalisation, puis effectuer le produit matriciel de cette matrice avec celle des demandes pour produire une nouvelle matrice de demandes directement exploitable par notre algorithme

# Bibliographie


[1] Gondran M, Minoux M. *Graphes et Algorithmes.*Eyrolles, 1995.

[2] Lévy G, *Algorithmique Combinatoire.* Dunod, 1994.

[3] Cisco systems et al, *Technologies des interconnexions réseaux.* Campus Press 2001.

[4] Peterson Larry L, Davie Bruce S. *Réseaux d'ordinateurs, une approche système.* Vuibert, 1998.

[5] Partridge Craig *Les réseaux gigabit.* Addison-Wesley. 1993

[6] Perlman Radia *Interconnexions : Ponts et routeurs.* Addison-Wesley. 1994

[7] Stevens W. Richard *TCP/IP illustré. Tome 1 : Les protocoles.* ITP, 1996

[8] Pujolle Guy *Les réseaux.*Eyrolles, 1998

[9] Comer Douglas *TCP/IP Architecture, Protocoles, Applications.* Dunod, 2001

[10] Hunt Craig *TCP/IP Administration de réseau.* O'relly, 1998

[11] Martins E. Q. V, Santos J. L. E. *The Optimal Path Problem.* Universidade de Coimbra Potugal 1999.

[12] Halabi Sam *OSPF Design Guide.* NSA group 1996

[13] Staehle D. ,kohler S. , Kohlhaas. U. *Towards an optimization of the routing parameters for IP networks.*

[14] Berry L., Kohler S., Staehle D., Tran-Gia P. *Fast heuristics for optimal routing in large Ip networks. Research report series,* report No 262, 2000.

[15] Milbrandt J., Kohler S., Staehle D., Berry L. *Decomposition of large IP networks for routing optimization.* Research report series, 2002.

[16] Dean H. Lorenz, Orda A., Raz D., Shavitt Y. *How good can IP routing be ?* DIMACS technical report, 2001

[17] Basak D., Kaur H. T., Kalyanaraman S. *Traffic engineering techniques and algorithms for the Internet.*2002

[18] Feamster A., Greenberg A. G. *Netscope traffic engineering for Ip networks.* IEEE network magazine, 2000

[19] Ericsson M., Resende, M.G.C., P.M. Pardalos P.M. *A genetic algorithm for the xeight setting problem in OSPF routing.* 2001

[20] Ramalingam G., Reps T. *An incremental algorithm for a generalization of the shortest path problem. Journal of algorithms,* 1996

[21] Ramalingam G., Reps T. *On the computational complexity of dynamic grah problems. Theoretical computer science,* 1996

[22] Frigioni D., Spaccamela A. M., Nanni U. *Fully dynamic algorithms for maintaining shortest path trees. Journal of algorithms,* 2000

[23] Frigioni D., Spaccamela A. M., Nanni U. *Experimental analysis of dynamic algorithms for single sourve shortest path problem. Journal of algorithms,* 1997

[24] Demetrscu D., Italiano G.F. *Fully dynamic all pairs shortest paths with real edge weights.*

[25] Harris, R. Kohler S. *Possibilities for Qos in existing Interent routing protocols, Research report series,* 1999

[26] Fortz B., Thorup M *Increasing Internet capacitty using local search. Technical report. Université libre de Bruxelle,* 2000

[27] Fortz B., Thorup M *Interent traffic engineering by optimizing OSPF weights. In proc. 19ht IEEE Conf.* INFOCOM 2000.

[28] Fortz B., Thorup M. *Optimizing OSPF/IS-IS weights in a changing world. IEEE JSAC,* Spring 2002

[29] Glover F. Laguna M. *Tabu search,* 1998

[30] Battati R. , Tecchiolli G. *The reactive Tabu search. ORSA journal on computing,* 1994.

[31] Taillard E.D. *Recherche itératives dirigées parallèles. Thèse de doctorat, EPFL,* 1993

[32] Bachelet V., Hafidi Z. Preux P, Tabli E. *Vers la cooperation des métaheuristiques,* 1998.

[33] Talbi E. *A taxonomy of hybrid metaheuristics. Journal of combinatorial optimization,* 1998

[34] Goldberg D.E. *Algorithmes génétiques. Exploration, optimisation et apprentissage automatique. Addison-Wesley* 1994

[35] Man K.F., Tang K.S. , Kwang S. *Genetic algorithms. Concepts and Designs.* Springer 1999

[36] Legault G. *Un algorithme génétique pour la conception topologique de réseaux téléinformatiques à commutation de paquets. Mémoire, Université du Quebec,* 1994.

[37] Barnier N. *Optimisation oar hybridation d'un algorithme génétique avec la programmation par contraintes.* DEA, Toulouse, 1997

[38] Sridharam A., Guerin R., Diot C. *Achieving near-optimal traffic engineering solution for current OSPF/IS-IS networks.* NANOG 27, Arizona 2001

[39] Nucci A et al. *IS-IS link weight assignment for transient link failures. Spint ATL, technical report,* 2002

[40] Ombuki B.M , Nakamura M , Osamu M. *A hybrid search based on genetic algorithms and tabu search for vehicule routing . Technical report,* 2002 Legout Arnaud *Contrôle de congestion multipoint pour les réseaux best Effort.*Thèse de Doctorat , Université de Nice , 2000.

[41] Cateloin Stéphane *Routage et qualité de service dans le réseau Internet* Thèse de Doctorat, Université de technologie de Compiègne , 2001

[42] Galinier P Hoa J. *Hybrid evolutionnary algorithms for graph coloring Journal of combinatorial optimization ,* 1999

[43] Dorne R Hoa J. *A new genetic local search algorithm for graph coloring Journal of combinatorial optimization ,* 1998

[44] Barichard V Hoa J. *Genetic tabu search for the multiobjective knapsack problem. Journal of combinatorial optimization*

[45] Rebreyend P. *Algorithmes génétiques hybrides en optimisation combinatoire.* Thèse de Doctorat, Université de Lyon, 1999

[46] Moy J. *OSPF specification.*RFC 1131, 1989

[47] Moy J. *OSPF version 2*RFC 1247, 1991

[48] Moy J. *OSPF protocol analysis.*RFC 1245, 1991

[49] Katz D. et al. *Traffic engineering extention to OSPF*RFC 2026, 2002.

[50] Awduche et al. *Overview and principles of Internet traffic engineering.*RFC 3272, 2002.

[51] Bettaher H. *Les protocoles de routage multipoints et la qualité de service dans l'Internet.* Thèse de Doctorat , Université de technologie de Compiègne , 2001

[52] Alvarez-Hamelin J.I. *Routage dans Internet : trafic autosimilaire, multicast et modèles de topologie.* Thèse de Doctorat , Université de Paris-sud, 2002.