Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Ferhat ABBAS – Sétif
UFAS - Algérie

THESE

Présentée à la faculté de Sciences de l'Ingénieur

Département
Informatique

pour l'obtention du diplôme de
Magistère

par
Mr. Nassir HARRAG

Thème

Optimisation des paramètres d'un réseau Ad Hoc par algorithmes génétiques

Soutenue publiquement le 21/09/2011 devant un jury composé de :

Mr.	M. Aliouat	M.C.A. à l'Université de Sétif	Président du jury
Mr.	A. Reffoufi	M.C.A. à l'Université de Sétif	Rapporteur
Mr.	A. Boukaram	M.C.A. à l'Université de Sétif	Examinateur

Remerciements

J’ajoute quelques lignes à ce document pour remercier ceux qui ont participé de près où de loin à sa naissance.

Je pense tout particulièrement au Docteur A. Boukaram pour avoir accepté de présider mon jury de thèse et au Docteur M. Aliouat d’avoir bien voulu disséquer les quelques 100 pages de cette thèse sans m’en tenir trop rigueur.

Merci également à Mes Frère Fouzi et Ghani pour leur aide.

Je suis reconnaissant envers le Docteur A. Reffoufi pour m’avoir guidé durant ces années. Je le remercie d’avoir su rester présent malgré ses nombreuses obligations.

J’accorde une place à part dans ces remerciements à tous mes amis et ceux qui ont contribué de près ou de loin à la naissance de ce travail.

Et puis merci à ceux qui étaient là dès le commencement. Merci Maman et Papa pour votre soutien et votre confiance. Merci à ma famille de m’avoir accompagné tout au long de mon cursus.

Résumé

Le problème de routage dans les réseaux Ad Hoc, en particulier le routage proactif, a suscité un vif intérêt dans la communauté des chercheurs. Bien que les protocoles proposés dans la littérature présentent certaines caractéristiques pertinentes, ils présentent certaines limites, surtout à forte mobilité des nœuds ou à forte charge du réseau qui dépendent des paramètres du réseau Ad Hoc choisis souvent d'une manière intuitive par un expert aguerri. Cette thèse détaille nos travaux pour solutionner cette tâche délicate et généralement manuelle utilisant un algorithme génétique pour l'automatisation de la sélection des paramètres. L'utilisation de l'algorithme génétique multi-objectif permet non seulement de minimiser le délai moyen de transfert du réseau et du nombre des paquets perdus mais en plus de s'adapter à chaque changement de topologie ce qui le rend adaptatif à tout environnement.

Tables des matières

Sommaire

Liste des tableaux

Liste des Tableaux

Liste des figures

Liste des Figures

Introduction

Depuis longtemps, les communications sans fil ont existé grâce à des signaux comme le feu, la fumée, des signes, mais aussi grâce au son, comme le langage sifflé. Le premier réseau de télécommunications sans fil de grande envergure est apparu en 1794 avec le Télégraphe Chappe utilisant un moyen de communication optique par sémaphore, étendant le réseau sur des centaines de kilomètres. Cependant en 1895, avec les expériences de Guglielmo Marconi, la transmission sans fil à l'aide d'ondes électromagnétiques fait son apparition et va devenir le meilleur moyen de transmettre de l'information via le télégraphe et le langage Morse.

Rapidement, les communications sans fil se sont développées dans de nombreux domaines comme l'aéronautique, les voitures, etc. Entre temps le réseau téléphonique filaire s'est répandu dans le monde et, depuis les années 1980, l'idée de pouvoir téléphoner partout à n'importe qui a donné naissance au réseau téléphonique mobile (ou radiotéléphone). Plus tard, avec l'augmentation de l'utilisation des ordinateurs personnels mobiles, les réseaux informatiques sans fil ont fait leur apparition, leur but étant de relier les ordinateurs dans un espace réduit (pièce, bâtiment). Rapidement, le besoin d'être connecté à Internet partout est devenu une problématique importante des réseaux informatiques sans-fil.

Les dernières années l'essort du besoin de plus de mobilité et de pouvoir d'accès à des données partagées ou échanger de l'information à tout moment, en utilisant des dispositifs mobiles (téléphones portables, PDA, PC portables,...) a répandu la notion de réseau sans infrastructure, ou réseaux Ad Hoc.

Un réseau informatique statique est composé de terminaux et routeurs. Un routeur permet de router les paquets, soit vers la destination finale, le terminal, soit vers un autre routeur. A l'opposé, un réseau Ad Hoc ne contient pas de routeurs dédiés : tout noeud peut être terminal et routeur. Par conséquent deux nœuds d'un réseau Ad Hoc peuvent communiquer sans la mise en place d'infrastructure comme dans un réseau classique. De plus, un réseau Ad Hoc est principalement utilisé dans un contexte de mobilité, alors que les protocoles de routage utilisés dans un réseau classique ne sont pas adaptés à un changement de topologie fréquent.

Dans un réseau Ad Hoc, deux nœuds d'un réseau s'échangent des données sous la forme de paquets à l'aide de nœuds appelés noeuds relais (ou noeuds intermédiaire). Pour cela, le noeud initiateur de l'échange, appelé noeud source, transmet les paquets à ses voisins. Si le noeud n'est pas la destination du paquet, un noeud relais transmet un paquet à ses voisins, et les voisins transmettent à leurs voisins, jusqu'à ce que le paquet ait atteint la destination. Lorsqu'un noeud reçoit un paquet, un protocole de routage prend la décision de, soit le transmettre à un noeud voisin, soit l'envoyer à la couche application, soit l'ignorer. Si le nœud est le destinataire de paquet, le paquet est envoyé à la couche application. S'il est un nœud relais pour la destination du paquet, alors il transmet le paquet. Sinon, le paquet est ignoré.

Dans un réseau Ad Hoc mobile, les noeuds se déplacent et donc la topologie change fréquemment. Un protocole adapté doit être utilisé et doit supporter la mobilité des noeuds. Chaque noeud exécute ce protocole pour calculer le prochain noeud relais. Suivant ce protocole, les noeuds échangent des messages de signalisation entre eux pour avoir connaissance de la topologie en créant un graphe du réseau ou pour créer un chemin vers la destination.

Le protocole de routage est indispensable pour connaître le chemin permettant d'atteindre la destination. Nous distinguons deux grandes classes de protocoles Ad Hoc: les protocoles géographiques et les protocoles topologiques. La première la première catégorie de protocoles, on trouve les protocoles géographiques pour lesquels chaque noeud doit connaître sa position géographique qui peut être obtenue à l'aide d'un système de géo-localisation tel que le GPS. Aucune connaissance globale du réseau (telle que la topologie) n'est nécessaire au routage des paquets : un noeud transmet un paquet au voisin le plus proche de la destination. De fait, ils sont bien adaptés aux changements fréquents de topologie du réseau. Par contre, la connaissance de la position géographique du noeud destinataire est indispensable et est fournie par un service de localisation associé au protocole Ad Hoc géographique.

La deuxième classe regroupe les protocoles Ad Hoc basés sur la topologie du réseau où les noeuds n'ont aucune connaissance de leur position géographique ni de celle des autres noeuds. Par échanges de messages entre voisins, ces protocoles doivent découvrir et maintenir la topologie du réseau pour router les paquets, soit de manière réactive soit manière proactive. Les travaux de cette thèse traitent cette dernière classe de protocole de routage.

Problématique et motivation

Le problème de routage dans les réseaux Ad Hoc, en particulier le routage proactif, a suscité un vif intérêt dans la communauté des chercheurs. Bien que les protocoles proposés dans la littérature présentent certaines caractéristiques pertinentes, ils présentent certaines limites, surtout à forte mobilité des nœuds ou à forte charge du réseau qui dépendent des paramètres du réseau Ad Hoc choisis souvent d'une manière intuitive par un expert aguerri. Cette thèse détaille nos travaux pour solutionner cette tâche délicate et généralement manuelle utilisant un algorithme génétique pour l'automatisation de la sélection des paramètres. L'utilisation de l'algorithme génétique multi-objectif permet non seulement de minimiser la latence du réseau et du nombre

des paquets perdus mais en plus de s'adapter à chaque changement de topologie ce qui le rend adaptatif à tout environnement.

Structure du document

Cette thèse est répartie en 5 chapitres. Le chapitre 1 donne un état de l'art du réseau Ad Hoc. Le chapitre 2 décrit les deux classes de méthodes d'optimisation à savoir les méthodes déterministes et les méthodes heuristiques. Les différents aspects d'un algorithme génétique sont détaillés dans le chapitre 3. Le chapitre 4 est dédié aux algorithmes génétiques multi-objectifs en mettant la lumière sur l'algorithme Non-dominated Sorting Genetic Algorithm NSGA-II. L'implémentation du NSGA-II ainsi que les résultats obtenus pour la sélection automatiques des paramètres sont présentés dans le chapitre 5. Enfin, la discussion des résultats et les perspectives concluent les travaux de cette thèse.

Chapitre

1 - Les réseaux mobiles Ad Hoc

Sommaire

Résumé

Ce chapitre propose une introduction aux réseaux mobiles Ad Hoc, leurs caractéristiques, leurs applications et les problèmes liés au routage. Il présente ensuite une classification des différents protocoles de routage dans ce type de réseaux.

1.1 Introduction

Aujourd'hui, les réseaux sans fil sont de plus en plus populaires du fait de leur facilité de déploiement. Ces réseaux jouent un rôle crucial au sein des réseaux informatiques. Ils offrent des solutions ouvertes pour fournir la mobilité ainsi que des services essentiels la où l'installation d'infrastructures n'est pas possible.

On peut classer les réseaux sans fil en trois catégories. Les premiers sont les réseaux sans fil centralisés, utilisés notamment dans la téléphonie mobile. Chaque téléphone communique avec un BTS (Base Transcription Service) qui gère les communications avec les autres mobiles. La deuxième catégorie est celle des réseaux ad-hoc. Ce sont des réseaux auto-organisés dans ce sens qu'il n'existe pas d'unité centrale (telle que les BTS) qui gère les communications. Cette catégorie se découpe elle-même en deux sous-catégories. La première est celle des réseaux ad-hoc single hop, où deux nœuds peuvent communiquer si et seulement si ils sont voisins directs. La deuxième sous-catégorie celle des réseaux ad-hoc proprement dit, où chaque nœud peut communiquer avec n'importe quel autre nœud du réseau (pourvu que celui-ci soit connexe) grâce à l'utilisation de protocoles de routage. Enfin, la dernière catégorie est celle des réseaux hybrides, où l'on étend un autre réseau par un réseau ad-hoc.

En général, un réseau Ad Hoc mobile (MANET : Mobile Ad Hoc Network) est considéré comme un système autonome dynamique composé de nœuds mobiles interconnectés par des liens sans fil, sans l'utilisation d'une infrastructure fixe et sans administration centralisée. Les réseaux ad-hoc ont la particularité de s'auto-créer, s'auto-organiser et s'auto-administrer et ne reposent sur aucune infrastructure fixe. Les éléments les composant sont en général reliés par radio, potentiellement mobiles, et peuvent être amenés à entrer ou sortir du réseau à tout moment. Ainsi, dans ce type de réseau, chaque nœud peut être à la fois client, serveur et routeur. Il est client lorsque qu'il désire envoyer un message à un autre nœud, serveur lorsqu'il effectue une opération pour un autre nœud, et routeur lorsqu'il aide à la transmission d'un message entre deux nœuds non voisins.

1.2 Les réseaux Ad Hoc

Un réseau mobile Ad Hoc (Ghosekar et al., 2010 ; Lemlouma, 2000), appelé généralement MANET (Mobile Ad Hoc NETwork), consiste en une grande population, relativement dense, d'unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces sans fil, sans l'aide d'une infrastructure préexistante ou administration centralisée.

Dans un réseau Ad Hoc, deux noeuds d'un réseau s'échangent des données sous la forme de paquets à l'aide de noeuds appelés noeuds relais (ou noeuds intermédiaire). Pour cela, le noeud initiateur de l'échange, appelé noeud source, transmet les paquets à ses voisins. Si le noeud n'est pas la destination du paquet, un noeud relais transmet un paquet à ses voisins, et les voisins transmettent à leur voisins, jusqu'à ce que le paquet ait atteint la destination Lorsqu'un noeud reçoit un paquet, un protocole de routage prend la décision de, soit le transmettre à un noeud voisin, soit l'envoyer à la couche application, soit l'ignorer. Si le nœud est le destinataire de paquet, le paquet est envoyé à la couche application. S'il est un noeud relais pour la destination du paquet, alors il transmet le paquet. Sinon, le paquet est ignoré.

1.3 Représentation des réseaux ad-hoc

Un réseau ad-hoc, tel que celui de la figure 1.1 est généralement représenté par un graphe G = (V,E) où V représente l'ensemble des nœuds du réseau et E l'ensemble des liens existant entre les différents nœuds. Ainsi, E est un ensemble de couplet (u, v) où u. Si un couplet (u, v) est dans E, alors u peut communiquer avec v.

Les réseaux Ad Hoc héritent des mêmes propriétés et problèmes liés aux réseaux sans fil. Particulièrement, le fait que le canal radio soit limité en terme de capacité, plus exposé aux pertes (comparé au médium filaire), et sujet à des variations dans le temps. Le canal est confronté aux problèmes de 'station cachée' et 'station exposée'. En outre, les liens sans fil sont asymétriques et pas sécurisés.

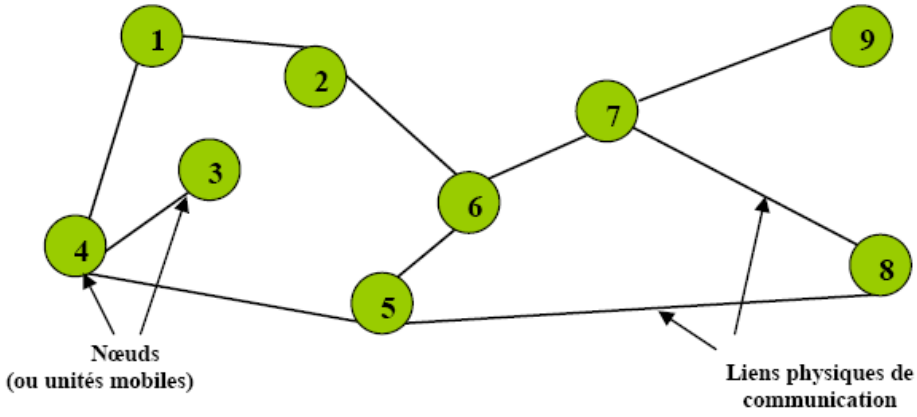

Figure 1.1 Modélisation d'un réseau Ad Hoc.

D'autres caractéristiques spécifiques aux réseaux Ad Hoc conduisent à ajouter une complexité et des contraintes supplémentaires qui doivent être prises en compte lors de la conception des algorithmes et des protocoles réseaux, à savoir (Ghosekar et al., 2010 ; Chlamtac, 2003):

- **L'absence d'une infrastructure centralisée**

  Chaque noeud travaille dans un environnement pair à pair distribué, et agit en tant que routeur pour relayer des communications, ou génère ses propres données. La gestion du réseau est ainsi distribuée sur l'ensemble des éléments du réseau ;

- **La mobilité des noeuds et maintenance des routes**

  La mobilité continue des nœuds crée un changement dynamique de topologie. Par exemple, un noeud peut joindre un réseau, changer de position ou quitter le réseau. Ce déplacement a naturellement un impact sur la morphologie du réseau et peut modifier le comportement du canal de communication. Ajoutons à cela la nature des communications (longues et synchrones, courtes et asynchrones,. . ..). Les algorithmes de routage doivent ainsi résoudre ces problèmes et supporter la maintenance et prendre en charge en un temps limité la reconstruction des routes tout en minimisant l'overhead généré parles messages de contrôle ;

- **L'hétérogénéité des nœuds**

  Un noeud mobile peut être équipé d'une ou plusieurs interfaces radio ayant des capacités de transmission variées et opérant dans des plages de fréquences différentes. Cette hétérogénéité de capacité peut engendrer des liens asymétriques dans le réseau. De plus, les noeuds peuvent avoir des différences en terme de capacité de traitement (CPU, mémoire), de logiciel, de taille (petit, grand) et de mobilité (lent, rapide). Dans ce cas, une adaptation dynamique des protocoles s'avère nécessaire pour supporter de telles situations ;

- **La contrainte d'énergie**

  Les équipements mobiles disposent de batteries limitées, et dans certains cas très limitées tels que les PDA, et par conséquent d'une durée de traitement réduite. Sachant qu'une partie de l'énergie est déjà consommée par la fonctionnalité du routage. Cela limite les services et les applications supportées par chaque nœud ;

- **Une bande passante limitée**

  Une des caractéristiques primordiales des réseaux basés sur la communication sans fil est l'utilisation d'un médium de communication partagé. Ce partage fait que la bande passante réservée à un hôte soit modeste ;

- **Sécurité et Vulnérabilité**

  Dans les réseaux Ad Hoc, le principal problème ne se situe pas tant au niveau du support physique mais principalement dans le fait que tous les nœuds sont équivalents et potentiellement nécessaires au fonctionnement du réseau. Les possibilités de s'insérer dans le réseau sont plus grandes, la détection d'une intrusion ou d'un déni de service est plus délicate. En plus, l'absence de centralisation pose un problème de remontée de l'information de détection d'intrusions.

En plus, les réseaux Ad-hoc souffrent de pas mal de lacunes comme les liaisons asymétriques (des communications à sens unique entre nœuds), le problème

d'interférences qui engendre un taux d'erreurs de transmission et affaiblit plus les performances d'un lien radio et la mobilité des nœuds qui entraîne des ruptures fréquentes des routes ce qui provoque un taux d'erreurs assez conséquent.

1.4 Domaines d'application des réseaux Ad Hoc

Les premières applications des réseaux Ad Hoc concernaient les communications et les opérations dans le domaine militaire. Cependant, avec l'avancement des recherches dans le domaine des réseaux et l'émergence des technologies sans fil (ex : Bluetooth, IEEE 802.11 et Hiperlan) ; d'autres applications civiles sont apparues (Ghosekar et al., 2010 ; Royer et Toh, 1999). On distingue :

- **Les services d'urgence:** opération de recherche et de secours des personnes, tremblement de terre, feux, inondation, dans le but de remplacer l'infrastructure filaire ;
- **Le travail collaboratif et les communications dans des entreprises ou bâtiments:** dans le cadre d'une réunion ou d'une conférence par exemple ;
- **Home network:** partage d'applications et communications des équipements mobiles ;
- **Applications commerciales:** pour un paiement électronique distant (taxi) ou pour l'accès mobile à l'Internet, où service de guide en fonction de la position de l'utilisateur ;
- **Réseaux de senseurs:** pour des applications environnementales (climat, activité de la terre, suivi des mouvements des animaux, … etc.) ou domestiques (contrôle des équipements à distance) ;
- **Réseaux en mouvement:** informatique embarquée et véhicules communicants ;
- **Réseaux Mesh:** c'est une technologie émergente qui permet d'étendre la portée d'un réseau ou de le densifier.

En plus, dans un LAN, un réseau Ad Hoc fournit une solution pour étendre une couverture sans fil avec un moindre coût. Dans un WAN (ex : UMTS), il permet d'accroître la capacité globale du réseau sans fil. En fait, plus de bande passante agrégée peut être obtenue en réduisant la taille des cellules et en créant des pico-cellules. Afin de supporter une telle architecture, les opérateurs disposent de deux options : déployer plus de stations de base (une station de base par cellule), ou utiliser un réseau Ad Hoc pour atteindre la station de base. La deuxième solution est clairement plus flexible et moins coûteuse.

1.5 Communication dans les réseaux Adhoc

Les signaux envoyés par les interfaces sans fil s 'atténuent au fur et à mesure qu'ils s'éloignent de leur émetteur, un noeud dans les réseaux Ad Hoc ne peut donc communiquer avec un autre s'il est situé trop loin de lui, il doit alors passer par des noeuds intermédiaires pour atteindre la destination désirée.

la figure 1.2 illustre la topologie d'un réseau Ad Hoc composé de cinq noeuds (A, B, C, D, E). Si le noeud A veut communiquer avec le noeud D, qui n'est pas dans sa porté de transmission, il doit passer par les noeuds intermédiaires B et E, on parle alors de communication « Multi-sauts » ou « Multi-hops ».

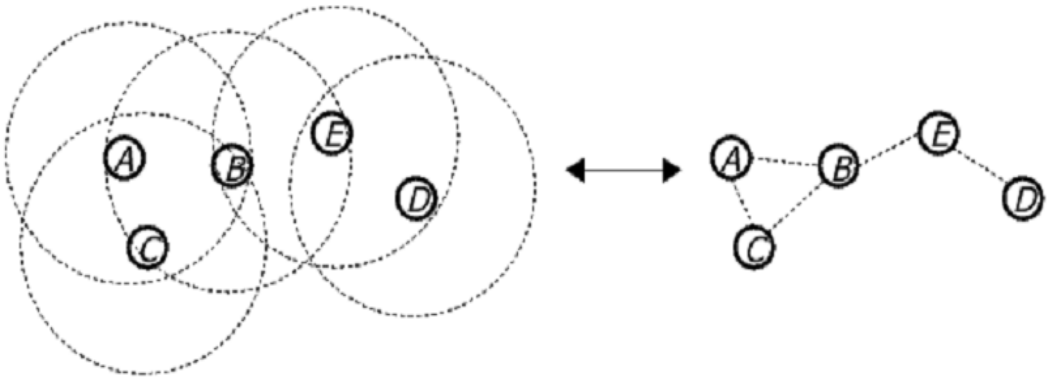

Figure 1.2 Communication dans les réseaux Ad Hoc.

Dans la figure 1.2 le noeud B peut communiquer directement avec les noeuds A, C et E, ces noeuds sont dis « Voisins ». En général, un paquet dans le réseau Ad Hoc doit traverser plusieurs noeuds avant d'atteindre sa destination.

La technique la plus basique pour réaliser une communication entre un noeud source et un noeud destination dans les réseaux Ad Hoc est l'inondation, ou le noeud source va transmettre les paquets à tous les noeuds voisins, chaque noeud mobile ré-émet a son tour les paquets qu'il reçoit à ses voisins jusqu'à ce qu'ils arrivent à la destination. L'inondation consomme beaucoup de ressources (bande passante et énergie) et n'est pas adaptée pour les réseaux Ad Hoc, d'où la nécessité d'avoir des protocoles de routage adaptés à ce type de réseaux.

1.6 Routage dans les réseaux Ad Hoc

Le routage est une méthode d'acheminement des informations vers la bonne destination à travers un réseau. Il consiste à assurer une stratégie qui garantit, à n'importe quel moment, d'établir des routes qui soient correctes et efficaces entre n'importe quelle paire de nœud appartenant au réseau, ce qui assure l'échange des messages d'une manière continue (Lemlouma, 2000).

A cause de la mobilité des noeuds dans les réseaux Ad Hoc, il est très difficile de localiser une destination à un instant donné, les protocoles de routage conçus pour des réseaux statiques sont inadaptés pour ce type de réseaux. Un protocole de routage pour les réseaux Ad Hoc doit organisé le réseau et prendre en compte les limitations existantes, il doit construire et maintenir les routes entre les différents noeuds et s'adapter à la topologie changeante et imprévisible.

Dans ce qui suit, nous décrirons brièvement la difficulté de routage dans les réseaux Ad Hoc et les différents mécanismes de routages apparus pour la résolution de ce problème.

1.6.1 Les contraintes de routages dans les réseaux Ad Hoc

L'accomplissement de la tâche de routage, inhérente à tout réseau, est rendue compliquée dans les réseaux Ad Hoc par l'utilisation de communications par radio: la radio est en effet le medium le plus hostile à la propagation de l'information, du fait notamment des interférences entre les signaux et la complexité du traitement du ces derniers. D'autre part, le routage Ad Hoc est aussi compliqué par la mobilité des éléments susceptibles d'acheminer le trafic (c'est-à-dire les utilisateurs eux-mêmes). N'ayant pas été prévus pour ces dernières complications, les algorithmes de routage classiques ne peuvent donc pas être utilisés tels quels. Il semble donc important que toute conception de protocole de routage pour les réseaux Ad Hoc doit prendre en compte les contraintes suivantes:

- **Minimisation de la charge du réseau:** l'optimisation des ressources du réseau renferme deux autres sous problèmes qui sont l'évitement des boucles de routage, et l'empêchement de la concentration du trafic autour de certains noeuds ou liens ;
- **Offrir un support pour pouvoir effectuer des communications multi-points fiables:** Le fait que les chemins utilisés pour router les paquets de données puissent évoluer, ne doit pas avoir d'incident sur le bon acheminement des données. L'élimination d'un lien, pour cause de panne ou pour cause de mobilité devrait, idéalement, augmenter le moins possible les temps de latence ;
- **Assurer un routage optimal:** La stratégie de routage doit créer des chemins optimaux et pouvoir prendre en compte différentes métriques de coûts (bande passante, nombre de liens, ressources du réseau,... etc.). Si la construction des chemins optimaux est un problème dur, la maintenance de tels chemins peut devenir encore plus complexe, la stratégie de routage doit assurer une maintenance efficace de routes avec le moindre coût possible ;
- **Le temps de latence:** La qualité des temps de latence et de chemins doit augmenter dans le cas où la connectivité du réseau augmente.

1.6.2 Classification des protocoles de routage

Il existe plusieurs critères pour la conception et la classification de protocoles de routage dans les réseaux ad-hoc : la manière dont les informations de routage sont échangées, quand et comment les routes sont calculées. Ainsi, il est possible de distinguer 3 grandes catégories de routage (Abolhasan et al., 2003 ; Elorrieta, 2007 ; Badache et al., 2002):

- **Protocoles proactifs:** ils établissent les routes à l'avance en se basant sur l'échange périodique de tables de routage ;
- **Protocoles réactifs:** ils recherchent les routes à la demande du réseau ;
- **Protocoles hybrides:** ils combinent les deux approches précédentes afin de tirer avantages de deux catégories précédentes, tout en réduisant leurs inconvénients.

Parmi ces 3 catégories de protocole de routage, certain ont une caractéristique supplémentaire. Il crée ou impose une hiérarchie sur le réseau. Cette approche consiste à « superposer » à la topologie physique, une topologie logique pour le routage.

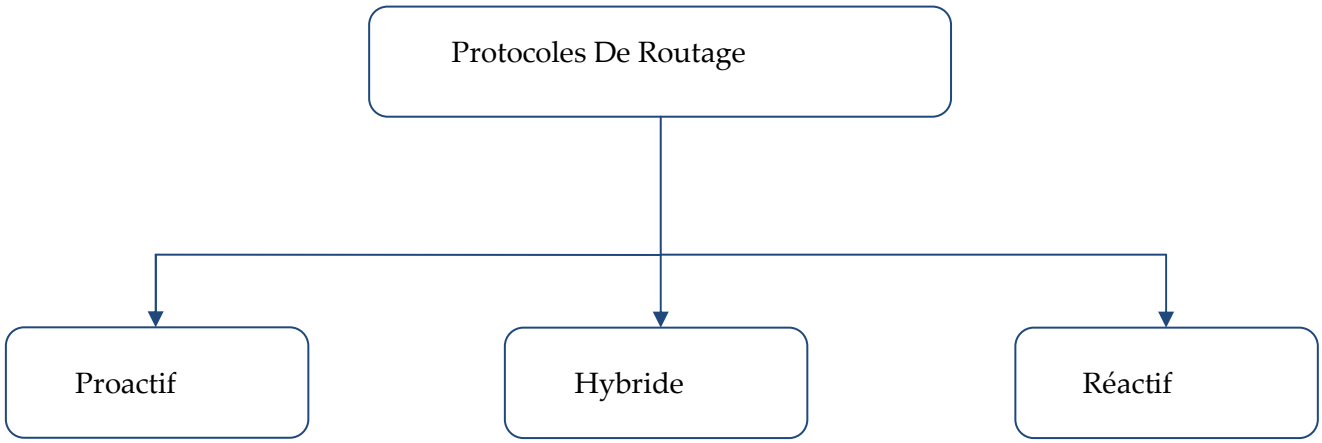

Figure 1.3 Classification des protocoles.

1.6.2.1 Les protocoles proactifs

Les protocoles proactifs calculent les routes à l'avance, en continu. Chaque noeud met à jour une ou plusieurs tables de routage par échange de paquets de contrôle entre voisins, de sorte que lorsqu'une application désire envoyer un paquet à un autre mobile la route est immédiatement connue (Théoleyre, 2006).

Dans les réseaux Ad Hoc les noeuds peuvent apparaître ou disparaître de manière aléatoire et la topologie même du réseau peut changée, cela signifie qu'il va falloir un échange continuel d'information pour que chaque noeud ait une image à jour du réseau. L'avantage premier de ce type de protocole est d'avoir les routes immédiatement disponibles quand les applications ont en besoin, le transfert des données sera plus rapide, mais cela se fait au coût d'échanges réguliers de messages (consommation de bande passante) qui ne sont certainement pas tous nécessaires (seules certaines routes seront utilisées par les applications en général).

Les principaux protocoles de cette famille est l'OLSR (Optimised Link State Routing) et DSDV (Destination Sequenced Distance-Vector Routing protocole).

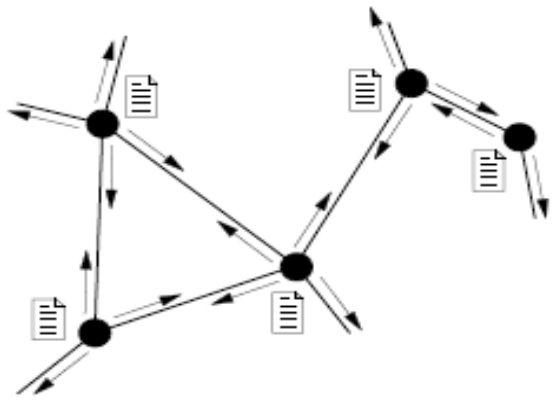

Figure 1.4 Maintien des tables de routage.

- **Destination Sequenced Distance Vector (DSDV)**

  Largement inspiré du protocole Routing Information Protocol (RIP) Destination Sequenced Distance-Vector Routing Protocol (Perkins et Bhagwat, 1994), DSDV est un des premiers protocoles proposés pour les réseaux Ad Hoc sans fil. Ce protocole est basé sur une version distribuée de l'algorithme de Bellman-Ford qui maintient une table contenant la longueur et le premier noeud du chemin vers chaque noeud du réseau. La table de routage est envoyée à tous les voisins à intervalle régulier afin de la maintenir à jour. Pour maintenir à jour la table de routage d'un réseau dynamique, l'échange de table complet à chaque changement de topologie implique la transmission d'un grand nombre de messages de signalisation. La taille de ces messages et leur fréquence d'envoi sont proportionnelles au diamètre du graphe représentant le réseau. En effet, plus le réseau est grand, plus la taille de la table de routage est grande, et plus il existe de liens dans le réseau, plus le nombre de changements d'état d'un lien du réseau est grand. Ce problème de passage à l'échelle, ou scalability en anglais, est fondamental dans les réseaux en général, et plus particulièrement dans les réseaux sans fil où la ressource radio est limitée ;

- **Optimized Link State Routing (OLSR)**

  Optimized Link State Routing Protocol (OLSR) (Clausen et Jacquet, 2008) tente de résoudre ce problème de passage à l'échelle. OLSR est un protocole à état de lien. Comme le protocole OSPF pour les réseaux filaires, le protocole maintient l'état de tous les liens du réseau pour construire le graphe du réseau et calculer le plus court chemin à l'aide d'un algorithme comme Dijkstra. OLSR utilise deux messages: Hello et Topology Control (TC). Le message Hello permet de découvrir les voisins et détecter le changement d'état d'un lien. Le message tc permet de propager une information de changement d'état d'un lien à l'ensemble des noeuds du réseau. Pour permettre le passage à l'échelle du protocole, chaque noeud sélectionne un sous-ensemble de ses voisins qu'il appellera "MPR" (MultiPoint Relais) pour retransmettre ses paquets en cas de diffusion. En se

basant sur la diffusion sur les MPRs, tous les noeuds du réseau sont atteints avec un nombre réduit de répétitions.

L'ensemble MPR est alors construit dans chaque noeud de façon à contenir un sous-ensemble de voisins à un saut qui couvre tous les voisins à deux sauts. Afin de construire les tables nécessaires au routage des paquets. Chaque noeud génère périodiquement un paquet TC contenant la liste de ses voisins l'ayant choisit comme MPR. Le message TC est diffusé dans l'ensemble du réseau, seuls les voisins MPR rediffusent un paquet TC reçu pour éviter l'inondation. Cette technique prometteuse réduit considérablement l'overhead généré par le trafic de contrôle. A la réception d'un message TC, la table de topologie peut être construite. Basé sur la table de topologie, chaque nœud peut calculer la table de routage qui permet d'acheminer les paquets vers n'importe quelle destination dans le réseau.

1.6.2.2 Les protocoles réactifs

Le principe des protocoles réactifs est de ne rien faire tant qu'une application ne demande pas explicitement d'envoyer paquet vers un noeud distant. Cela permet d'économiser de la bande passante et de l'énergie. Lors qu'un paquet doit être envoyé le protocole de routage va chercher un chemin jusqu'à la destination. Une fois ce chemin trouvé il est inscrit dans la table de routage et peut être utilisé .En générale cette recherche se fait par inondation (un paquet de recherche de route est transmit de noeud en noeud dans tout ou une partie du réseau, après réception de ce paquet par le noeud cible il renvoi un paquet réponse qui va remonter vers la source et ainsi tracer le chemin pour l'échange des données). L'avantage majeur de cette méthode est qu'elle ne génère du trafic de contrôle que lors qu'il est nécessaire. Les principales contre parties sont que l'inondation est un mécanisme coûteux qui va faire intervenir tous les noeuds du réseau ou une partie et qu'il va y avoir un délai à l'établissement des routes. Les principaux protocoles de cette famille sont DSR, AODV:

- **Dynamic Source Routing (DSR)**

Le protocole DSR (Johnson et al., 2007) est basé sur l'utilisation de la technique du routage par la source. Dans cette technique la source détermine la séquence complète des nœuds à travers lesquels des paquets de données seront envoyés. Avant d'envoyer un paquet de données vers un autre nœud, l'émetteur diffuse un paquet « route request ». Si l'opération de découverte de routes est réussie, l'émetteur reçoit un paquet « route response » qui contient une séquence de nœuds à travers laquelle la destination peut être atteinte. Le paquet « route request » contient un champ d'enregistrement de routes, dans lequel sera accumulée la séquence de nœuds visités durant la propagation de la requête dans le réseau. L'utilisation de la technique de routage par la source fait que les nœuds de transit n'ont pas besoin de maintenir les informations de mise à jour pour envoyer les paquets de données, puisque ces derniers contiennent toutes les décisions de routage.

- **Ad Hoc On-demand Distance Vector (AODV)**

Ce protocole représente essentiellement une amélioration de l'algorithme DSR (Perkins et al., 2003). Il réduit le nombre de diffusions de messages et cela en créant les routes lors du besoin, contrairement à DSR qui maintient la totalité des routes. Il maintient ces routes aussi longtemps que cela est nécessaire au besoin des nœuds sources. AODV utilise un numéro de séquence afin de dater les différentes routes pour avoir des routes à jour. En raison de la mobilité des nœuds dans les réseaux Ad Hoc, les routes changent fréquemment ce qui fait que les routes maintenues par certains nœuds deviennent invalides. Les numéros de séquence permettent d'utiliser les routes les plus récentes.

1.6.2.3 Les protocoles hybrides

Les protocoles hybrides combinent les deux approches. Ils utilisent un protocole proactif, pour apprendre le proche voisinage (voisinage à deux ou trois sauts) et un protocole réactif pour atteindre les noeuds situés au-delà de cette zone prédéfinie de voisinage. Les protocoles hybrides font appels aux techniques des protocoles réactifs

pour chercher des routes. Avec ce découpage, le réseau est partagé en plusieurs zones et la recherche de route en mode réactif peut être améliorée. A la réception d'une requête de recherche réactive, un noeud peut indiquer immédiatement si la destination est dans le voisinage ou non et par conséquent savoir s'il faut aiguiller ladite requête vers les autres zones sans déranger le reste de sa zone. Ce type de protocole s'adapte bien aux grands réseaux, cependant, il accumule aussi les inconvénients des protocoles réactifs et proactifs : messages de contrôle périodique coût de découverte d'une nouvelle route.

- **Zone Routing Protocole (ZRP)**

  Zone routing protocole (ZRP) (Haas, 1997) est un protocole de routage Ad Hoc hybride qui combine les avantages d'un protocole réactif avec les avantages d'un protocole proactif. L'idée est d'utiliser un routage proactif à l'intérieur d'une zone limité à r-hop de tout noeud et d'utiliser un routage réactif au delà de cette zone. Un intra-zone routing protocole (IARP) est utilisé dans la zone où le routage est proactif et un inter-zone routing protocole (IERP) est utilisé au delà. IARP peut être n'importe quel protocole Ad Hoc proactif et IERP n'importe quel protocole Ad Hoc réactif.

- **Zone Based Hierarchical (ZHLS)**

  Le protocole ZHLS (Joa-Ng et Lu, 1999) est basé sur la décomposition d'un réseau en zones. Contrairement à la plupart des protocoles dit hiérarchiques, il n'y a pas ici de représentant pour chaque zone. La topologie d'un réseau est ainsi partagée en deux niveaux:

  * Un niveau nœud indique la façon dont les nœuds d'une zone sont connectés entre eux physiquement. Un lien virtuel peut exister entre deux zones s'il existe au moins un nœud d'une autre zone.
  * Un niveau zone qui renseigne sur le schéma de connexion des différentes zones.

  Ces niveaux différents entraînent donc deux différents types de liens: les liens inter-nœuds et les liens inter-zones.

1.7 Conclusion

Le concept des réseaux mobiles Ad Hoc permet d'étendre les notions de la mobilité à toutes les composantes de l'environnement. Le problème de routage, en particulier, a suscité un vif intérêt dans la communauté des chercheurs. Plusieurs travaux de recherche se sont intéressés au problème de routage dans les réseaux Ad Hoc. Bien que les protocoles proposés présentent certaines caractéristiques pertinentes, ils présentent certaines limites, surtout à forte mobilité des nœuds ou à forte charge du réseau. Dans cette thèse nous utiliserons un algorithme génétique afin d'automatiser la sélection des valeurs des paramètres Ad Hoc et donc minimiser les effets d'inondations massives tout en assurant une meilleure adaptation aux changements fréquents de topologie.

Chapitre

2 - Les méthodes d'optimisation

Sommaire

Résumé

Ce chapitre introduit le lecteur aux méthodes d'optimisation. Il l'initie aux concepts de base et terminologie concernant l'optimisation. Il expose également les deux grandes familles des méthodes d'optimisations, à savoir : 1) les méthodes déterministes, et 2) les méthodes non-déterministes.

2.1 Introduction

Les chercheurs et les ingénieurs sont confrontés quotidiennement à des problèmes de complexité grandissante, qui surgissent dans des secteurs très divers : comme dans le traitement des images, la conception de systèmes mécaniques, la recherche opérationnelle, la conception de systèmes informatiques … etc. le problème à résoudre peut souvent être considéré comme un problème d'optimisation dans lequel on définit une fonction objectif, ou fonction de coût(voire plusieurs) avec ou sans contraintes, que l'on cherche à optimiser par rapport à tous les paramètres concernés. Dans ce chapitre Nous n'aborderons pas le problème de l'optimisation d'un point de vue mathématique, mais simplement du point de vue d'un ingénieur pragmatique.

2.2 Essai -Erreur

La méthode de base pour optimiser un problème est la méthode d'*essai et erreur*, où la solution passe par diverses étapes d'expérimentation potentielles jusqu'à obtention de résultats adéquats. C'est ce que nous faisons quand nous donnons à un paramètres plusieurs valeurs successives et que nous observons le résultat. C'est également le cas quand nous faisons varier ce paramètre de façon quasi-continue à l'aide d'un ordinateur et que nous traçons la courbe correspondante.

D'un point de vue pratique, le processus d'optimisation se décompose en 3 étapes :

- L'analyse ;
- La synthèse ;
- L'évaluation.

..

La figure 2.1 illustre le processus d'optimisation.

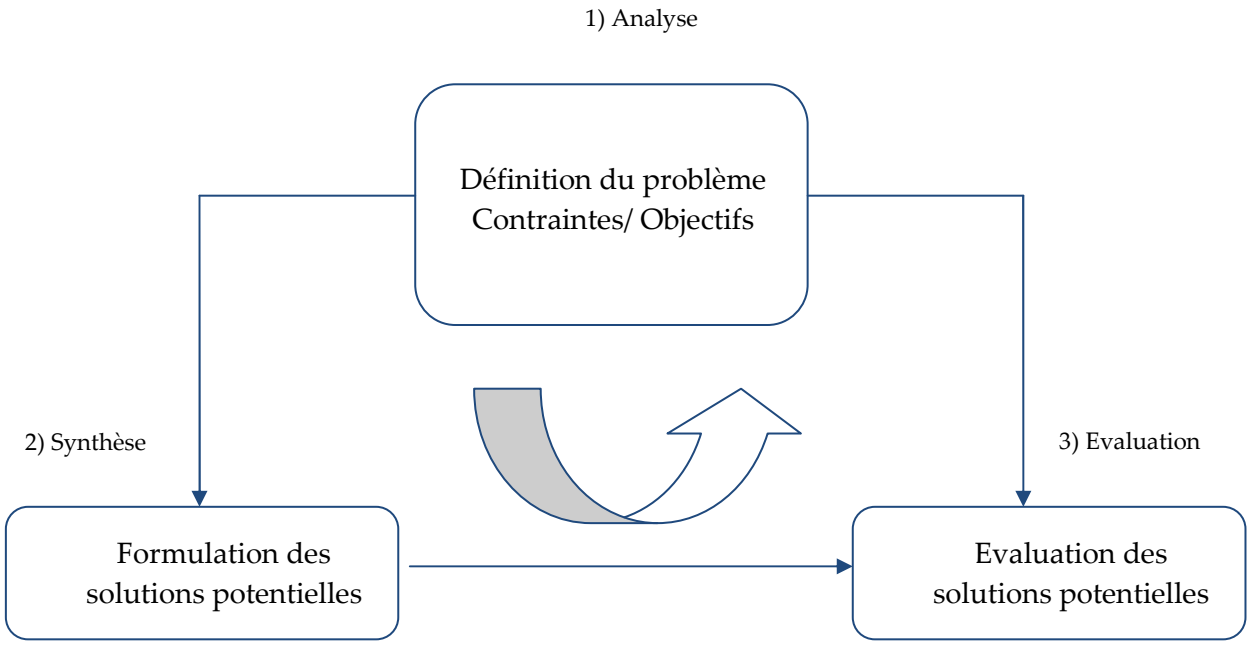

Figure 2.1 Processus d'optimisation selon Asimow.

L'analyse du problème est en substance la réponse à un certain nombre de choix préalables :

- **Variables du problème :** Quels sont les paramètres intéressants à faire varier ?
- **Espace de recherche:** Dans quelles limites faire varier ces paramètres ?
- **Fonctions objectif:** Quels sont les objectifs à atteindre ?
- **Méthode d'optimisation:** Quelle méthode choisir ?

La synthétisation de la méthode choisie, permet l'évolution de la solution trouvée voire son élimination jusqu' à obtention d'un résultat acceptable. Si nécessaire, le problème peut être redéfini à partir des solutions déjà obtenues.

2.3 Concepts de base et terminologie concernant l'optimisation

Tout d'abord, nous définirons les concepts communs à n'importe quelle méthode d'optimisation :

2.3.1 Variables du problème

Correspondent aux paramètres de la fonction objective. Ils sont ajustés pendant le processus d'optimisation, pour obtenir le(s) solution(s) optimale(s). On les appelle aussi variables d'optimisation, variables de conception ou de projet (design variables).

2.3.2 Espace de recherche

Défini par l'ensemble des combinaisons des valeurs des paramètres. Il correspond à l'espace des solutions. La dimension de l'espace de recherche est définie par le nombre de paramètres impliqués dans les solutions (par exemple, si chaque solution est définie par trois paramètres, l'espace de recherche est tridimensionnel). On l'appelle aussi espace des paramètres.

2.3.3 Fonction objectif

C'est une équation mathématique qui représente ce qu'on désire améliorer dans un problème. Elle est aussi appelée critère d'optimisation, fonction coût, fonction d'adaptation, ou encore performance (fitness fonction) ;

- **Objectif unique**

  Dans le cas d'un objectif unique, la définition de la fonction ne pose généralement pas de problème. Par exemple, si l'on se fixe l'objectif de trouver un dispositif dont le rendement est maximum, la fonction sera égale au rendement. Dans le cas où l'on utilise un modèle numérique, on commence par

évaluer les caractéristiques des solutions potentielles en utilisant le modèle. Puis on calcule la fonction d'adaptation à partir de ces caractéristiques ;

- **Objectifs multiples**

Certains problèmes d'optimisation doivent satisfaire des objectifs multiples, souvent concurrents, ce qui implique un compromis. La méthode classique consiste à définir plusieurs fonctions d'objectif f_i , traduisant chaque objectif à atteindre, et à les combiner au sein de la fonction d'adaptation. La combinaison la plus simple est une somme pondérée des fonctions objectif :

$$f = \sum_i \propto_i f_i$$

où $\propto_i$ sont les poids de normalisation de tel sorte à avoir toutes les valeurs des fonctions d'adaptation comprises dans l'intervalle [0 ; 1].

Remarquons qu'un poids peut être négatif afin de tenir compte de certaines contraintes du problème. C'est à l'utilisateur de fixer convenablement les poids. On peut généralement classer les objectifs par importance mais les poids devront être adaptés par tâtonnement, jusqu'à l'obtention d'une solution acceptable. Malgré l'automatisation du processus d'optimisation, l'utilisateur doit paramétrer à la main la définition de la fonction d'adaptation.

Il faut néanmoins être conscient des effets d'une telle combinaison des objectifs. En effet, deux solutions potentielles dont les fonctions d'objectifs n'ont pas la même valeur peuvent aboutir à une même valeur de la fonction d'adaptation. De plus, un algorithme utilisant une telle approche ne convergera que vers une seule solution alors qu'il existe peut-être toute une famille de solutions remplissant les objectifs fixés. L'optimisation à objectifs multiples est un domaine de recherche en dynamique actuellement, face aux enjeux économiques et industriels. Des concepts tels que les niches écologiques ou l'optimalité de Pareto semblent prometteurs pour la résolution de ce genre de problème.

2.3.4 Méthodes d'optimisation

Une fois définie la fonction à optimiser, le choix d'une méthodologie adaptée s'impose pour la résolution du problème. Les méthodes d'optimisation, répondent à deux classements distincts (Berro, 2001) : les *méthodes déterministes* et les *méthodes non-déterministes.* Les méthodes déterministes sont généralement utilisées quand l'évaluation de la fonction est très rapide, ou quand la forme de la fonction est connue a priori. Les cas plus complexes (temps de calcul important, nombreux optima locaux, fonctions non-dérivables, fonctions fractales, fonctions bruitées…) seront souvent traités plus efficacement par des méthodes non-déterministes.

- **Les méthodes déterministes**

  L'efficacité des méthodes déterministes s'articule autour de la rapidité de l'évaluation de la fonction ou selon la connaissance à priori de ladite fonction. La recherche des extrema d'une fonction f revient à résoudre un système de n équations à n inconnues, linéaire ou non :

  $$\frac{\partial f}{\partial X_i}(x1, \dots\dots\dots\dots\dots, xn) = 0$$

  Ces méthodes est nécessite une étape de discrétisation de l'espace de recherche pour la localisation des extrema. La fonction à optimiser est évaluée en chacun des points de discrétisation. La valeur maximale est alors considérée comme une bonne approximation de l'optimum de la fonction. Cette méthode est brutale et le temps de calcul augmentera exponentiellement en fonction du nombre de variables.

  En générale, les méthodes déterministes sont très efficaces sur des problèmes particuliers et en générale de petite taille. Mais sur des problèmes de grande taille, la probabilité de trouvé l'optimum en un temps raisonnable dépend essentiellement de la bonne connaissance du problème par le décideur.

- **Méthodes non déterministes ou stochastiques (méta-heuristiques)**

  Ces méthodes sont caractérisées par :

  - Un processus de création aléatoire ou pseudo-aléatoire des points dans l'espace de recherche ;

  - Une heuristique qui permet de guider la recherche ;

  En générale, Les méthodes stochastiques sont utilisées dans des problèmes ou on ne connait pas d'algorithme de résolution en temps polynomiale et pour lesquels on espère trouver une solution approchée de l'optimum globale.

2.3.5 Méthodes déterministes

Les méthodes d'optimisation déterministes se caractérisent par une exploration systématique de l'espace de recherche. Parmi les nombreuses méthodes déterministes on peut distinguer deux approches : Locale et Globale. Les méthodes locales assurent la convergence vers l'optimum de la fonction le plus proche de la solution courante en explorant son voisinage alors que les méthodes globales s'attachent à faire converger la solution vers l'optimum global de la fonction.

D'une manière générale ces méthodes obéissent à l'algorithme suivant :

a) Choix d'une première solution courante i admissible ;

b) Génération d'une solution j dans le voisinage de i ;

c) Si $f(j)$ est meilleur que $f(i)$ alors j devient la solution courante puis retour en b) ;

d) L'algorithme se termine lorsqu'il n'y a plus d'amélioration de la solution courante.

Parmi les méthodes déterministes, on peut citer la méthode du gradient ou la méthode de Gauss-Seidel (Minoux, 1983). Ces Méthodes présentent un certain nombre d'inconvénients. Tout d'abord, elles nécessitent le choix des paramètres qui supposent

la disposition a priori de certaines informations liées au problème et ces informations ne se limitent pas à des priorités, une connaissance plus approfondie de la nature du problème est souvent nécessaire afin de prévoir comment tel ou tel choix des paramètres de la méthode va influer sur le résultat. Ainsi les méthodes déterministes ne fournissent qu'une seule solution, qui dépend fortement des paramètres choisis au départ. Dans le cas où les conditions citées ci-dessus ne sont pas réunies, le décideur devra s'orienter vers des méthodes non déterministes.

2.3.6 Méthodes non-déterministes

2.3.6.1 Recuit simulé

Le recuit simulé est une méthode d'optimisation apparue en 1982, avec la publication dans la revue science d'un article de Kirpatrick (Kirpatrick et al., 1983). Elle se fonde sur une analogie entre les problèmes d'optimisation et ceux de la physique statistique. L'application principale du recuit simulé est la recherche opérationnelle c'est-à-dire tous les problèmes d'optimisation où l'on désire maximiser ou minimiser une quantité. Le recuit simulé a déjà obtenu d'excellents résultats dans divers domaines ce qui a incité les mathématiciens, les physiciens et les ingénieurs à s'intéresser à cette méthode et à la développer.

Le recuit simulé fut obtenu par analogie avec le phénomène thermodynamique de recuit des métaux. Initialement, le métal est porté à très haute température, puis il est refroidi progressivement (Salomo, 2001):

- à haute température les atomes sont très agités de telle sorte que toutes les configurations atomiques sont équiprobables,
- à basse température les atomes s'organisent pour aboutir à une structure atomique parfaite proche de l'état d'énergie minimale.

Le refroidissement doit être très lent pour ne pas rester bloqué dans un minimum local : à chaque instant le métal doit se trouver dans un état de quasi-équilibre thermique permettant de sortir du minimum local en atteignant un autre état

d'équilibre associé à la température courante. Si le refroidissement est trop rapide, on obtient une étape qui au sens mathématique est une solution sous optimale.

Les premiers à avoir l'idée d'utiliser cet algorithme pour résoudre un problème d'optimisation furent Kirpatrick (Kirpatrick et al., 1983), ils remplacèrent la fonction d'énergie par la fonction à minimiser, une configuration atomique par une solution admissible du problème et introduisent d'autre part un schéma de température évoluant d'une valeur haute vers une valeur basse. Bien qu'originellement le recuit simulé fût introduit pour des problèmes discrets, il peut être utilisé pour résoudre des problèmes continus en discrétisant l'espace de recherche.

2.3.6.2 Les algorithmes de colonies de fourmis

Cette approche, due à Dorigo et ses collaborateurs (Dorigo al., 1991), s'efforce de simuler la capacité collective de résolution de certains problèmes, observée chez une colonie de fourmis dont les membres sont pourtant individuellement dotés de facultés très limitées. Apparues sur Terre il y a quelque 100 millions d'années, les fourmis sont en effet l'une des espèces les plus prospères : 10 millions de milliards d'individus, répartis partout sur la planète : leur masse totale est du même ordre de grandeur que celle des humains ! Le succès de cette espèce soulève de nombreuses questions. En particulier, les entomologistes ont analysé la collaboration qui s'établit entre les fourmis pour aller chercher de la nourriture à l'extérieur de la fourmilière : il est remarquable que les fourmis suivent toujours le même chemin, et que ce chemin est le plus court possible. Cette conduite est le résultat d'un mode de communication indirecte, via l'environnement : la stigmergie. Chaque fourmi dépose, le long de son chemin, une substance chimique, la phéromone ; tous les membres de la colonie perçoivent cette substance, et orientent préférentiellement leur marche vers les régions les plus « odorantes ». Il en résulte notamment la faculté collective de retrouver rapidement le plus court chemin, si celui-ci se trouve fortuitement obstrué par un obstacle. En s'inspirant de la modélisation de ce comportement, Dorigo et ses

collaborateurs ont proposé un nouvel algorithme pour la résolution du célèbre problème du voyageur de commerce.

Les algorithmes à colonies de fourmis fournissent de très bons résultats comparables en qualités à ceux obtenus avec les autres méta-heuristiques. Néanmoins, le principal inconvénient de l'approche réside dans le coût relativement élevé de la génération des solutions. Les temps de calcul pour une itération de l'algorithme à fourmis nous ont conduit à réaliser un très faible nombre d'itérations par rapport au nombre d'itérations réalisées par l'algorithme génétique par exemple. Cette grande différence dans le nombre d'itérations explique partiellement la dominance de l'algorithme génétique par rapport à l'algorithme de fourmis.

2.3.6.3 La recherche tabou

La méthode de recherche tabou, ou simplement méthode tabou, a été formalisée en 1986 par (Glove , 1986). Sa principale particularité tient dans la mise en œuvre de mécanismes inspirés de la mémoire humaine. La méthode tabou prend, sur ce plan, le contre-pied du recuit simulé, totalement dépourvu de mémoire, et donc incapable de tirer les leçons du passé. Le principe de cette méthode est à chaque itération le voisinage de la solution courante est examiné. L'algorithme enregistre la meilleure solution parmi les voisins, même si elle est moins bonne que la solution courante.

L'acceptation des solutions moins performantes que la solution courante permet d'éviter de tomber dans un optimum local. Pour échapper de tourner dans un cercle entre plusieurs solutions, l'algorithme interdit le passage par des solutions récemment visitées. En pratique la méthode stocke dans une liste taboue T les attributs des dernières solutions visitées. Dans l'itération suivante, la meilleure nouvelle solution voisine enlève la solution la plus ancienne dans la liste. Dans d'autres cas, la méthode mémorise les mouvements réalisés plutôt que les solutions. Ensuite, on interdit les mouvements inverses. Cette technique est rapide et consomme peu de mémoire.

2.3.6.4 Algorithmes Génétiques

Les algorithmes génétiques (AG) sont des techniques de recherche inspirées par l'évolution biologique des espèces. Introduits par J.H. Holland au début des années 1970 (Holland, 1975), ils ont d'abord suscité un intérêt limité, du fait de leur important coût d'exécution. Ils connaissent, depuis les années 1990, un développement considérable, notamment suite à apparition des architectures massivement parallèles, qui exploitent leur « parallélisme intrinsèque ».le principe d'un AG se décrit simplement, dans le cas de la minimisation d'une fonction f. Un ensemble de N points, qui peuvent être choisis au hasard, constitue la population initiale ; chaque individu x de la population possède une certaine compétence, qui mesure son degré d'adaptation à l'objectif visé : ici, x est d'autant plus compétent que f (x) est petit. Un AG consiste à faire évoluer progressivement, par générations successives, la composition de cette population, en maintenant sa taille constante. D'une génération à la suivante, la « compétence » de la population doit globalement s'améliorer ; un tel résultat est obtenu en mimant les deux principaux mécanismes qui régissent l'évolution des êtres vivants : la sélection naturelle (qui détermine quels membres d'une population survivent et se reproduisent) et la reproduction (qui assure le brassage et la recombinaison des gènes parentaux, pour former des descendants aux potentialités nouvelles).

En pratique, chaque individu est codé par une chaîne de caractères de longueur donnée (de même qu'un chromosome est formé d'une chaîne de gènes). Le passage d'une génération à la suivante se déroule en deux phases : une phase de reproduction et une phase de remplacement. La phase de reproduction consiste à appliquer des opérateurs, dits génétiques, sur les individus de la population courante, pour engendrer de nouveaux individus ; les opérateurs les plus utilisés sont le croisement, qui produit deux descendants à partir de deux parents, et la mutation, qui produit un nouvel individu à partir d'un seul individu. Les individus de la population prenant part à la reproduction sont préalablement sélectionnés, en respectant le principe suivant : plus un individu est compétent, plus sa probabilité de sélection est élevée. La phase de remplacement consiste ensuite à choisir les membres de la nouvelle

génération : on peut, par exemple, remplacer les plus « mauvais » individus (au sens de la fonction objectif) de la population courante par les meilleurs individus produits (en nombre égal). L'algorithme est interrompu après un nombre donné de générations.

2.4 Conclusion

Ce chapitre donne un tour d'horizon des différentes méthodes d'optimisation, à savoir : les méthodes déterministes et les méthodes non-déterministes. Il expose les deux familles d'algorithmes en mettant l'accent sur les algorithmes les plus répandus. Il introduit également l'optimisation heuristique plus particulièrement les algorithmes génétiques qui font l'objet de ces travaux de thèse et qui seront approfondis dans ce qui suit.

Chapitre

3 - Les algorithmes génétiques

Sommaire

Résumé

Ce chapitre détaille les concepts fondamentaux d'un algorithme génétique. Il introduit le principe, le codage et les différents opérateurs génétiques. Il donne également un exemple pour illustrer ces différents concepts.

3.1 Introduction

Les algorithmes génétiques sont des algorithmes d'optimisation s'appuyant sur des techniques dérivées de la génétique et de l'évolution naturelle : croisements, mutations, sélection, etc.

De l'origine des espèces En publiant L'Origine des espèces, en 1859, Charles Darwin (Darwin, 1859) expose pour la première fois sa théorie de l'évolution, désormais communément admise. Celle-ci repose sur les observations suivantes :

- il existe au sein de chaque espèce de nombreuses variations, chaque individu étant différent ;
- les ressources naturelles étant finies, il naît rapidement plus d'être vivants que la nature ne peut nourrir ; il en résulte une lutte pour l'existence (struggle for life) entre chaque organisme ;
- les individus survivants possèdent des caractéristiques qui les rendent plus aptes à survivre ; Darwin baptise ce concept sélection naturelle ;
- les organismes survivants transmettent leurs avantages à leur descendance ; l'accumulation au cours des générations des petites différences entre chaque branche généalogique crée de nouvelles espèces de plus en plus aptes à survivre.

Ces quatre principes de la théorie darwinienne ont été étendus dans les années 1930 pour incorporer les découvertes en génétique. Cette théorie synthétique de l'évolution a été baptisée néo-darwinisme et correspond aux bases de l'approche en vigueur dans la communauté scientifique. Dans le langage génétique moderne, les molécules d'ADN, constituées d'une suite de nucléotides sont le support du génotype – l'ensemble des gènes – dont l'expression physiologique est dénommée le phénotype. Il n'existe pas de définition précise de la notion de gène qui soit universellement reconnue. Sans entrer dans les détails, on peut considérer qu'un gène est une séquence de nucléotides codant pour un caractère particulier, le plus souvent une protéine.

Réaliser ce fantasme sur un ordinateur afin de créer des machines plus « intelligentes » anime Turing dès 1948. Mais il faut attendre les travaux autour des stratégies d'évolution au milieu des années 1960 puis ceux de John H. Holland sur les algorithmes génétiques peu de temps après pour observer les premières utilisations pratiques appliquant les concepts du néo-darwinisme à l'informatique.

3.2 Définition

Les algorithmes génétiques sont une abstraction sur la théorie d'évolution (Goldberg, 1989). L'idée fondatrice est simple : si l'évolution a optimisé les processus biologique, l'utilisation du paradigme de l'évolution pour trouver des solutions optimales dans le cadre de traitement informatiques possède un sens.

L'utilisation de la théorie de l'évolution comme modèle informatique pour trouver une solution optimale peut se justifier par le fait que la théorie d'évolution permet la recherche la solution parmi un très grand de possibilités dans un laps de temps raisonnable.

3.3 Applications

Les applications des AG sont multiples : optimisation de fonctions numériques difficiles (discontinues, multimodales, bruitées…), traitement d'image (alignement de photos satellites, reconnaissance de suspects…), optimisation d'emplois du temps, optimisation de design, contrôle de systèmes industriels, apprentissage des réseaux de neurones, etc. Les AG peuvent être utilisées pour contrôler un système évoluant dans le temps (chaîne de production, centrale nucléaire…) car la population peut s'adapter à des conditions changeantes. En particulier, ils supportent bien l'existence de bruit dans la fonction à optimiser. Ils peuvent aussi servir à déterminer la configuration d'énergie minimale d'une molécule ou à modéliser le comportement animal.

Les AG sont également utilisées pour optimiser des réseaux (câbles, fibres optiques, mais aussi eau, gaz…), des circuits VLSI , des antennes… Ils peuvent être utilisés pour trouver les paramètres d'un modèle petit-signal à partir des mesures expérimentales. Des commutateurs optiques adiabatiques ont été optimisés à l'aide des Stratégies d'Evolutions (autres AE) chez SIEMENS AG. On envisage l'intégration d'AG dans certaines puces électroniques afin qu'elles soient capables de se reconfigurer automatiquement en fonction de leur environnement (Evolving Hardware en anglais) .

3.4 Principe

Un algorithme génétique recherche le ou les extrema d'une fonction définie sur un espace de données. Pour l'utiliser, on doit disposer des cinq éléments suivants :

- **Un principe de codage de l'élément de population:** Cette étape associe à chacun des points de l'espace d'état une structure de données. Elle se place généralement après une phase de modélisation mathématique du problème traité. Le choix du codage des données conditionne le succès des algorithmes génétiques. Les codages binaires ont été très employés à l'origine. Les codages réels sont désormais largement utilisés, notamment dans les domaines applicatifs, pour l'optimisation de problèmes à variables continues ;
- **Un mécanisme de génération de la population initiale:** Ce mécanisme doit être capable de produire une population d'individus non homogène qui servira de base pour les générations futures. Le choix de la population initiale est important car il peut rendre plus ou moins rapide la convergence vers l'optimum global. Dans le cas où l'on ne connaît rien du problème à résoudre, il est essentiel que la population initiale soit répartie sur tout le domaine de recherche ;
- **Une fonction à optimiser:** Celle-ci prend ses valeurs dans R et est appelée fitness ou fonction d'évaluation de l'individu. Celle-ci est utilisée pour sélectionner et reproduire les meilleurs individus de la population ;

- **Des opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace d'état:** L'opérateur de croisement recompose les gènes d'individus existant dans la population, l'opérateur de mutation a pour but de garantir l'exploration de l'espace d'état ;
- **Des paramètres de dimensionnement:** taille de la population, nombre total de générations ou critère d'arrêt, probabilités d'application des opérateurs de croisement et de mutation.

Le principe général du fonctionnement d'un algorithme génétique est représenté sur la figure 3.1, L'algorithme commence avec un ensemble de solutions possibles du problème (individus), constituant une population. Les individus sont formés par des variables, qui sont les paramètres à ajuster dans un dispositif (par exemple, la longueur et la largeur du système à optimiser). Cette population est conçue aléatoirement à l'intérieur de limites prédéfinies (par exemple, les limites dictées par les aspects constructifs).

Certaines solutions de la première population sont utilisées pour former, à partir d'operateurs génétiques (croisement, mutation, etc.), une nouvelle population. Ceci est motivé par l'espoir que la nouvelle population soit meilleure que la précédente. Les solutions qui serviront à former de nouvelles solutions sont sélectionnées aléatoirement d'après leurs mérites (représentés par une « fonction objectif » spécifique au problème posé, qui devra être minimisée ou maximisée) : meilleur est l'individu, plus grandes seront ses chances de se reproduire (c'est-à-dire, plus grande sera sa probabilité d'être sélectionné pour subir les opérateurs génétiques). Ceci est répété jusqu'à ce qu'un critère de convergence soit satisfait (par exemple, le nombre de générations ou le mérite de la meilleure solution).

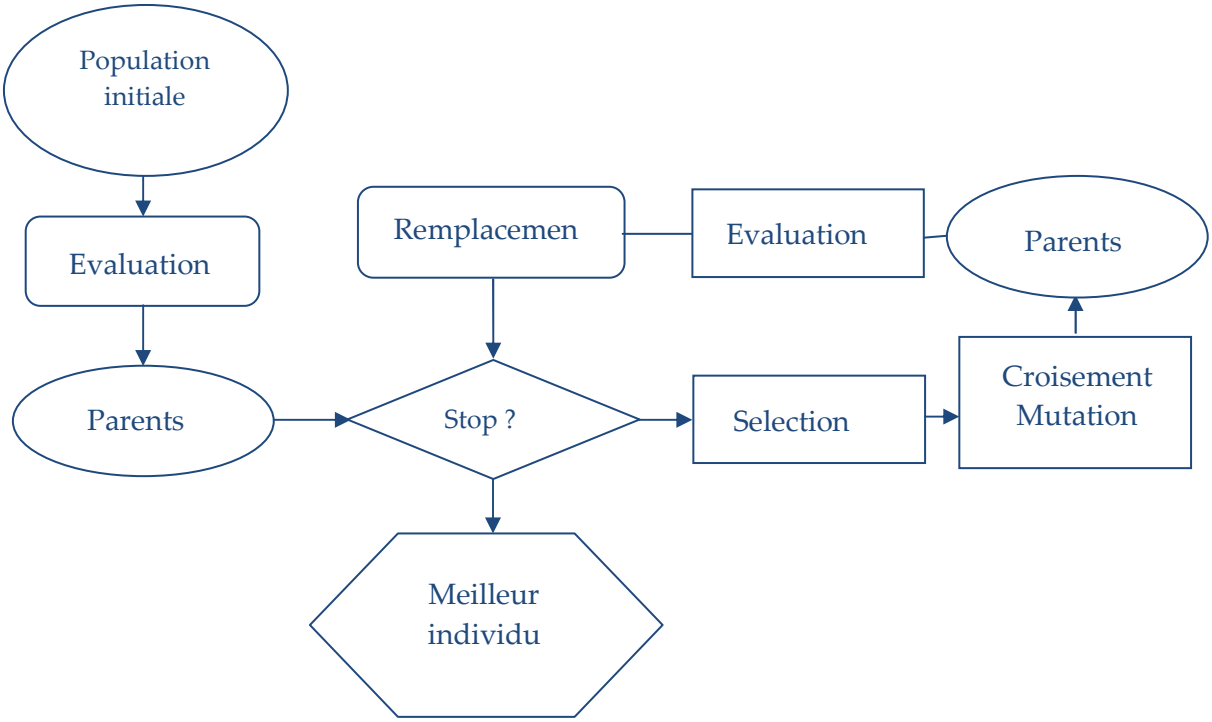

Figure 3.1 Squelette d'un algorithme génétique.

3.5 Codage et population initiale

Premièrement, il faut représenter les différents états possibles de la variable dont on cherche la valeur optimale sous forme utilisable pour un AG: c'est le codage. Cela permet d'établir une connexion entre la valeur de la variable et les individus de la population, de manière à imiter la transcription génotype-phénotype qui existe dans le monde vivant. Il existe principalement deux types de codage : le codage binaire, le codage réel.

3.5.1 Codage binaire

Ce codage a été le premier à être utilisé dans le domaine des AG. Il présente plusieurs avantages: alphabet minimum {0,1}, facilité de mise en point d'opérateurs génétiques et existence de fondements théoriques (théorie sur les schémas). Néanmoins ce type de codage présente quelques inconvénients :

- Les performances de l'algorithme sont dégradées devant les problèmes d'optimisation de grande dimension à haute précision numérique. Pour de tels problèmes, les AG basés sur les chaînes binaires ont de faibles performances comme le montre michalewicz(Michalewicz, 1992).
- La distance de Hamming entre deux nombres voisins (nombre de bits différents) peut être assez grande dans le codage binaire : l'entier 7 correspond à la chaîne 0111 et la chaîne 1000 correspond à l'entier 8. Or la distance de hamming entre ces deux chaînes est de 4, ce qui crée bien souvent une convergence, et non pas l'obtention de la valeur optimale.

1	0	0	1

Figure 3.2 Représentation binaire.

3.5.2 Codage réel

Il a le mérite d'être simple. Chaque chromosome est en fait un vecteur dont les composantes sont les paramètres du processus d'optimisation. Par exemple, si on recherche l'optimum d'une fonction de n variables $f(x_1 1, x_2, x_3, x_4, \ldots, x_n)$, on peut utiliser tout simplement un chromosome ch contenant les n variables: Avec ce type de codage, la procédure d'évaluation des chromosomes est plus rapide vu l'absence de l'étape de transcodage (du binaire vers le réel). Les résultats donnés par Michalewicz (Michalewicz, 1992) montrent que la représentation réelle aboutit souvent à une meilleure précision et un gain important en termes de temps d'exécution.

x_1	x_2	.....	x_n

Figure 3.3 Représentation réel.

3.6 Génération aléatoire de la population initiale

Le choix de la population initiale d'individus conditionne fortement la rapidité de l'algorithme. Si la position de l'optimum dans l'espace d'état est totalement inconnue, il est naturel d'engendrer aléatoirement des individus en faisant des tirages uniformes dans chacun des domaines associés aux composantes de l'espace d'état, en veillant à ce que les individus produits respectent les contraintes (Michalewicz et Janikov, 1991). Si par contre, des informations a priori sur le problème sont disponibles, il paraît bien évidemment naturel d'engendrer les individus dans un sous-domaine particulier afin d'accélérer la convergence. Dans l'hypothèse où la gestion des contraintes ne peut se faire directement, les contraintes sont généralement incluses dans le critère à optimiser sous forme de pénalités.

3.7 Opérateurs de sélection

La sélection permet d'identifier statistiquement les meilleurs individus d'une population et d'éliminer les mauvais. On trouve dans la littérature un nombre important de principes de sélection plus ou moins adaptés aux problèmes qu'ils traitent.

La manière la plus classique d'implémenter l'opérateur de sélection est la méthode de la roulette biaisée. Ce principe se base sur l'image d'une roulette de casino telle que la probabilité de sélectionner un individu particulier soit proportionnelle à la valeur de sa fonction d'adaptation. Ainsi, les meilleurs individus auront une probabilité plus importante d'être sélectionnés pour la reproduction. Cependant, des tests empiriques ont montré qu'il était plus efficace de se baser sur le classement des individus dans la population, appelé leur rang, plutôt que sur la valeur de la fitness.

Dans ce dernier cas, si un individu obtient une fitness nettement meilleure que les autres, il risque d'envahir rapidement la population. Cette réduction rapide de la diversité génétique mène facilement à une convergence prématurée vers un unique minimum local.

Pour sélectionner les individus en fonction de leur rang, on commence par les classer en fonction de leur fitness par ordre décroissant puis on définit la probabilité de tirer un rang i comme le multiple de la probabilité de tirer le rang du dessous par un facteur multiplicatif. On aboutit à la formule suivante, permettant de tirer un index i :

$$i = n - \frac{n}{\log(k+1)} . \log\left(k . rand(1) + 1\right)$$

où k = c^{n+1}- 1, n est le nombre d'individus que l'on souhaite prendre en compte, c un coefficient par exemple égal à 1, 1 et rand(1) un générateur aléatoire renvoyant un nombre réel compris entre 0 et 1.

Une autre méthode classique de sélection est celle du tournoi. On tire successivement des couples d'individus et on sélectionne celui qui possède la meilleure fitness. On itère le processus jusqu'à ce que l'on ai sélectionné suffisamment de parents.

Ces deux approches peuvent être combinées au principe de l'élitisme. Afin d'éviter la perte des meilleurs individus, on peut choisir de les recopier directement vers la génération suivante. On évite ainsi que la meilleure fitness puisse décroître et que les très bonnes solutions soient éliminées par la nature stochastique (aléatoire) de l'opérateur de sélection. On conserve généralement 5 à 10 % des meilleurs individus.

Comme ce sera le cas pour les autres opérateurs, il existe très peu de base théorique pour guider le choix de la méthode de sélection. Seuls des tests concernant votre application spécifique vous permettront de déterminer l'approche la plus efficace.

3.8 Opérateurs de croisement

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants.

Initialement, le croisement associé au codage par chaînes de bits est le croisement à découpage de chromosomes (slicing crossover). Pour effectuer ce type de croisement sur des chromosomes constitués de M gènes, on tire aléatoirement une position dans chacun des parents. On échange ensuite les deux sous-chaînes évolutionnaires comme montre figure 3.4.

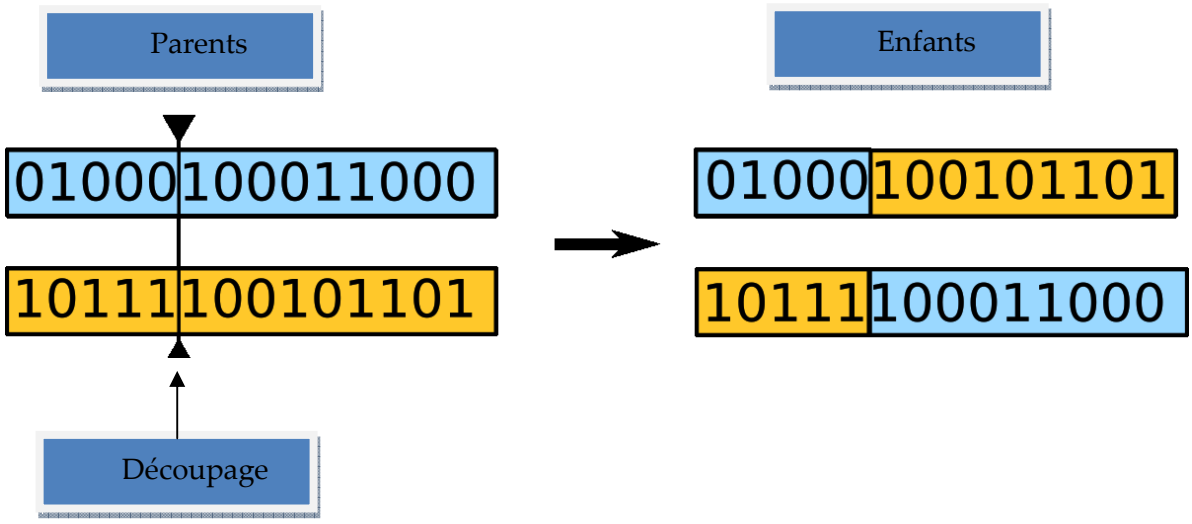

Figure 3.4 Croisement par découpage sur une chaîne binaire.

Ce type de croisement à découpage de chromosomes est très efficace pour les problèmes discrets. Pour les problèmes continus, un croisement « barycentrique » est souvent utilisé : Pour générer deux enfants e1 et e2 à partir des parents p1 et p2, on commence par tirer un nombre α au hasard dans l'intervalle [−0.5; 1.5], on applique ensuite les formules :

$$e1 = \alpha\ p1 + (1 - \alpha)p2$$

$$e2 = \alpha\ p2 + (1 - \alpha)p1$$

3.9 Opérateur de mutation

L'opérateur de mutation apporte aux algorithmes génétiques la propriété d'ergodicité de parcours d'espace. Cette propriété indique que l'algorithme génétique sera susceptible d'atteindre tous les points de l'espace d'état, sans pour autant les parcourir

tous dans le processus de résolution. Ainsi en toute rigueur, l'algorithme génétique peut converger sans croisement, et certaines implémentations fonctionnent de cette manière. Les propriétés de convergence des algorithmes génétiques sont donc fortement dépendantes de cet opérateur sur le plan théorique.

La définition de la mutation dans le cadre du codage binaire est particulièrement simple. Dans un premier temps, une ou plusieurs positions sont tirées. Le ou les bits correspondants sont lors remplacés par une valeur tirée aléatoirement (en l'occurrence 0 ou 1).

Comme dans le cas du croisement, cette approche fonctionne mal sur des nombres réels. La stratégie la plus courante est d'ajouter du bruit à certains membres du vecteur, tirés aléatoirement. Différentes distributions de probabilité, centrées sur la valeur du gène avant la mutation, peuvent être envisagées. Dans le cas d'une loi uniforme, une variation est choisie uniformément dans un intervalle fixé autour de la valeur actuelle du gène. Une distribution gaussienne peut aussi être utilisée.

La largeur de l'intervalle choisi dans le cas de la loi uniforme et le paramètre d'écart type dans celui de la loi gaussienne peuvent être délicats à choisir. Si celui-ci est trop faible, l'espace de recherche risque ne pas être suffisamment exploré. À l'inverse, s'il est trop important, l'algorithme peut avoir des difficultés à affiner les solutions alors qu'il est proche de l'optimum. Une approche intéressante pour résoudre ce problème est de coder ces paramètres dans le génotype. Ainsi, les solutions explorant rapidement l'espace seront favorisées au début de l'algorithme et auront naturellement tendance à s'éteindre au fur et à mesure que le processus converge vers l'optimum.

3.10 Gestion des contraintes

Un élément de population qui viole une contrainte se verra attribuer une mauvaise fitness et aura une probabilité forte d'être éliminé par le processus de sélection.

Il peut cependant être intéressant de conserver, tout en les pénalisant, les éléments non admissibles car ils peuvent permettre de générer des éléments admissibles de

bonne qualité. Pour de nombreux problèmes, l'optimum est atteint lorsque l'une au moins des contraintes de séparation est saturée, c'est-à-dire sur la frontière de l'espace admissible.

Gérer les contraintes en pénalisant la fonction fitness est difficile, un « dosage » s'impose pour ne pas favoriser la recherche de solutions admissibles au détriment de la recherche de l'optimum ou inversement.

Disposant d'une population d'individus non homogène, la diversité de la population doit être entretenue au cours des générations, afin de parcourir le plus largement possible l'espace d'état. C'est le rôle des opérateurs de croisement et de mutation.

3.11 Un exemple simple

Nous reprenons ici l'exemple de Goldberg (Goldberg, 1989). Il consiste à trouver le maximum de la fonction f (x)=x^2 sur l'intervalle [0;31] où x est un entier. La première étape consiste à coder la fonction. Par exemple, nous utilisons un codage binaire de x, la séquence (chromosome) contenant au maximum 5 bits. Ainsi, nous avons x = 2{0,0,0,1,0}, de même x = 31 {1,1,1,1,1}. Nous recherchons donc le maximum d'une fonction de fitness dans un espace de 32 valeurs possibles de x.

3.11.1 Tirage et évaluation de la population initiale

Nous fixons la taille de la population à N = 4. Nous tirons donc de façon aléatoire 4 chromosomes sachant qu'un chromosome est composé de 5 bits, et chaque bit dispose d'une probabilité $\frac{1}{2}$ d'avoir une valeur 0 ou 1.

Nous observons que le maximum 16, est atteint par la deuxième séquence. Voyons comment l'algorithme va tenter d'améliorer ce résultat.

Tableau 3.1 Principe de fonctionnement d'un algorithme génétique

Numéro	Séquence	Fitnesse	% du Total
1	00101	5	14.3
2	10000	16	45.7
3	00010	2	5.7
4	00110	12	34.3
Total		**35**	**100**

3.11.2 Sélection

Une nouvelle population va être créée à partir de l'ancienne par le processus de sélection de la roue de loterie biaisée.

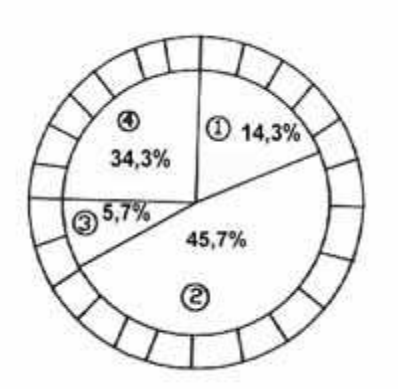

Figure 3.5 La roue de loterie biaisée : opération de sélection.

Nous tournons cette roue 4 fois et nous obtenons au final la nouvelle population

Tableau 3.2 Sélection par roue de loterie

Numéro	Séquence
1	10000
2	01100
3	00101
4	10000

3.11.3 Le croisement

Les parents sont sélectionnés au hasard. Nous tirons aléatoirement un lieu de croisement dans la séquence. Le croisement s'opère alors à ce lieu avec une probabilité pc. Le tableau suivant donne les conséquences de cet opérateur en supposant que les chromosomes 1 et 3, puis 2 et 4 sont appariés et qu'à chaque fois le croisement s'opère (par exemple avec pc = 1).

Tableau 3.3 Principe du crossover

l=3	l=2
100|00	01|100
001|01	10|000
10001	**01000**
00100	**10100**

3.11.4 La mutation

Dans cet exemple à codage binaire, la mutation est la modification aléatoire occasionnelle (de faible probabilité) de la valeur d'un bit (inversion d'un bit). Nous tirons ainsi pour chaque bit un chiffre aléatoire entre 0 et 1 et si ce chiffre est inférieur à pm alors la mutation s'opère. Le tableau suivant, avec pm = 0,05, met en évidence ce processus.

Tableau 3.4 Mutation des chromosomes

Chromosome	Tirage Aléatoire	Nouveau Bit	Nouveau Chromosome
10001	15 25 36 **04** 12	1	100**1**1
00100	26 89 13 48 59	-	00100
01000	32 45 87 22 65	-	01000
10100	47 **01** 85 62 35	1	**1**1100

Maintenant que la nouvelle population est entièrement créée, nous pouvons de nouveau l'évaluer.

3.11.5 Retour à la phase d'évaluation

Le maximum est maintenant de 28 (séquence 4). Nous sommes donc passé de 16 à 28 après une seule génération. Bien sûr, nous devons recommencer la procédure à partir de l'étape de sélection jusqu'à ce que le maximum global, 31, soit obtenu, ou bien qu'un critère d'arrêt ait été satisfait.

Tableau 3.5 Nouvelle génération

Numéro	Séquence	Fitnesse	% du Totale
1	10011	19	32.2
2	00100	4	6.8
3	01000	8	13.5
4	11100	28	47.5
Total		**59**	**100**

3.12 Conclusion

Ce chapitre est une bonne entrée en la matière du domaine des algorithmes génétiques plus particulièrement les algorithmes génétiques mono-objectif. Il sert d'un tremplin au chapitre suivant qui traite les algorithmes génétiques multi-objectif, ces algorithmes qui seront détaillés dans le chapitres 4 et qui seront utilisés et feront l'état des différentes expériences du chapitres 5.

Chapitre

4 - Les algorithmes génétiques multi-objectifs

Sommaire

Résumé

Ce chapitre approfondit les bases des méthodes d'optimisations en dressant les bases de l'optimisation génétique multi-objectifs. Il expose le problème multi-objectif en donnant les différents algorithmes proposés en mettant l'accent sur le NSGA-II.

4.1 Introduction

Les algorithmes génétiques utilisent une procédure d'optimisation basée sur l'emploi d'une population de solutions possibles au problème donné, en vue de rechercher la solution optimale. C'est pourquoi ils conviennent parfaitement pour traiter les problèmes d'optimisation multi objectif qui sont caractérisés, non pas par une solution unique, mais par un ensemble de solutions optimales, appelées solutions Pareto optimales.

4.2 Problème d'optimisation multi-objectif

Dans un problème d'optimisation multi objectif, on ne cherche pas d'optimiser une fonction objective unique mais un vecteur de fonctions :

$$\left\{ \underset{X \in C}{Min} \left(F(x) = (f_1(x), f_2(x), f_3(x), f_4(x), \ldots\ldots, f_n(x))\right)\right\}$$

où :

- n est le nombre de fonctions objectifs,
- X = [x_1 , x_2, ….. , x_m] est le vecteur représentant les variables de décision,
- C représente l'ensemble des solutions réalisables associé à des contraintes .
- et $F(x) = (f_1(x), f_2(x), f_3(x), f_4(x), \ldots\ldots, f_n(x))$ est le vecteur d'objectifs.

Dans un problème d'optimisation multi-objectif, il y a plus qu'une fonction d'objectif ($n \geq 2$), chaque fonction peut avoir un optimum différent. Le but d'un problème multi- objectifs est donc de trouver de "bons compromis" plutôt qu'une seule solution (à moins qu'une solution soit optimale pour toutes les fonctions d'objectifs, ce qui est rarement le cas). Lorsqu'il y a plusieurs objectifs, la notion d'optimum change, elle est remplacée par les notions de dominance et d'optimalité de Pareto.

4.2.1 La dominance

Une solution A domine une solution B pour un problème de minimisation (resp. maximisation) si et seulement si:

$$\forall\, i \in \{1,2,3,4,\ldots,n\}:\ f_i(A) \leq f_i(B)\ \left(resp\ f_i(A) \geq f_i(B)\right)$$

$$\exists j \in \{1,2,3,4,\ldots,n\}: f_j(A) < f_j(B)\ \left(resp\ f_j(A) > f_j(B)\right)$$

On dit que B est dominée par A ou entre les deux solutions, A est la solution non dominée.

4.2.2 Optimum de Pareto

Un vecteur $x^* \in C$ est un optimum de Pareto s'il n'existe aucune solution X de C qui domine x^*. Au lieu d'une solution unique, l'optimisation multi-objectifs donne lieu à un ensemble de solutions optimales. Toute solution de cet ensemble est optimale dans le sens qu'il est impossible d'améliorer les performances, sur un critère de cette solution, sans que cela entraîne une dégradation des performances, sur au moins un autre critère. Ces solutions optimales forment l'ensemble de solutions Pareto optimales, elles sont aussi connues sous le nom de *solutions efficaces*, *non inférieures* et *non dominées*. La représentation de ces solutions non dominées dans l'espace d'objectif est appelée front de Pareto.

La figure 4.1 montre un exemple de front de Pareto pour un problème de minimisation à deux objectifs. L'ensemble de points blancs représentent le front de Pareto.

Les algorithmes génétiques, avec un bon réglage de leurs paramètres, constituent une approche intéressante pour la résolution des problèmes d'optimisation multi-objectifs. De plus, ce domaine est très dynamique et ne cesse de se développer. Il a été proposé plusieurs méthodes pour le traitement des problèmes multi objectifs. Ces méthodes peuvent être classées principalement en deux catégories (Berro, 2001): La première catégorie enveloppe les méthodes "agrégatives", qui transforment le

problème en un problème uni-objectif utilisant une fonction objectif équivalente. La deuxième famille comporte les méthodes dites "non agrégatives", dans ces méthodes il n'y a pas fusion d'objectifs pour se ramener à un problème d'optimisation mono-objectif.

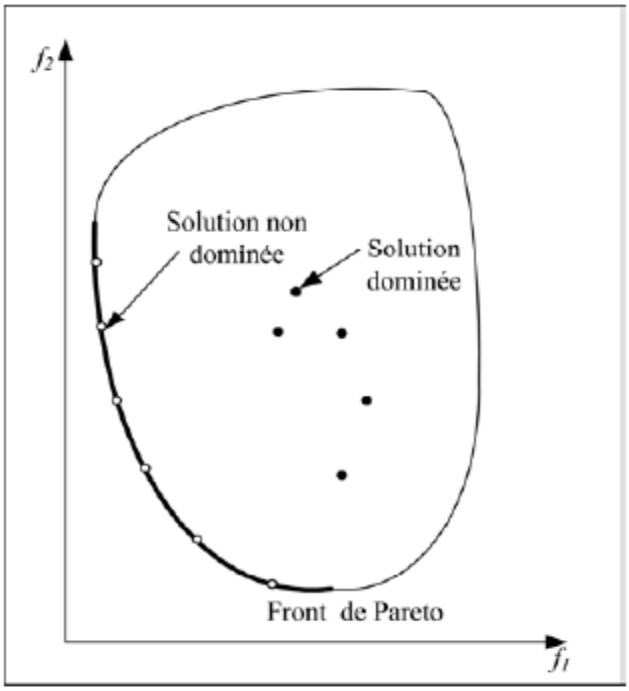

Figure 4.1 Front de Pareto.

4.2.2.1 Méthode agrégative

C'est l'une des premières méthodes utilisée pour résoudre les problèmes d'optimisation multi-objectifs (MO). Elle consiste à transformer le problème MO en un problème mono-objectif en combinant les composantes fi du vecteur objectif du problème en une seule fonction scalaire f. Il existe dans la pratique, différentes façons de construire la fonction f. La plus classique et la plus utilisée se ramène à une simple somme pondérée des objectifs fi (agrégation additive):

$$f = \sum_i \alpha_i \ f_i$$

où les paramètres α_i sont les poids de pondération.

4.2.2.2 *Méthode non agrégative*

Comme nous l'avons signalé précédemment, la méthode agrégative peut être utilisée de façon séquentielle pour obtenir le front de Pareto pour un problème d'optimisation multi-objectifs. Toutefois, cette approche n'est généralement pas satisfaisante car le nombre d'exécutions successives nécessaires pour déterminer les différents compromis conduit alors à un nombre d'évaluations de critères prohibitif. Ainsi, pour surmonter cette difficulté, on préfère utiliser des méthodes permettant d'une part de trouver l'ensemble de solutions Pareto optimales en une seule exécution et d'autre part de s'affranchir des problèmes de mise à l'échelle des objectifs. Parmi ces méthodes on peut citer :

- Vector Evaluated Genetic Algorithm (VEGA)
- Multiple Objective Genetic algorithm (MOGA)
- Niched Pareto genetic algorithm (NPGA)
- Nondominated sorting genetic algorithm (NSGA) et sa deuxième version (NSGA-II)

4.3 Etat de l'art des algorithmes évolutifs multi-objectifs

Les premiers algorithmes évolutifs multi-objectifs qui ont été développés sont non-élitistes , c'est à-dire qu'ils n'utilisent pas le principe d'archivage pour conserver les solutions Pareto optimales durant le processus d'optimisation. Ces algorithmes comprennent entre autres le MOGA (Multi- Objective Genetic Algorithm) de Fonseca et Fleming (Fonseca et Fleming, 1993), le NSGA (Non-dominated Sorting Genetic Algorithm) de Srinivas et Deb (Deb et Agrawal, 1994), et le NPGA (Niched-Pareto Genetic Algorithm) de (Horn et al., 1994) Ils mettent en œuvre le concept d'optimalité de Pareto pour classer les solutions en différents groupes de solutions non dominées, en vue d'aider à la convergence des algorithmes. Pour assurer la diversité de la population dans la région de Pareto, ces algorithmes utilisent la technique de la *fonction de partage* (Goldberg, 1989) dont on sait que le choix du paramètre de réglage σ_{share}

conditionne les performances des algorithmes. Ces algorithmes ne sont plus utilisés à cause de leur convergence lente et la difficulté de maintenir adéquatement une bonne diversité dans la population.

Plus tard, une autre génération d'algorithmes évolutifs multi-objectifs a vu le jour, et intègrent le concept d'archivage dans leur fonctionnement. Il s'agit entre autres de l'algorithme SPEA (Strength Pareto Evolutionary Algorithm) de Zitzler et Thiele qui utilise une méthode de regroupement des solutions (appelée "*clustering*"). C'est une méthode qui permet de distribuer efficacement les solutions sur la frontière de Pareto. Cependant, pour des problèmes présentant plusieurs fronts de Pareto locaux, le *clustering* conduit à un élitisme extrême. Knowles et Corne ont proposé l'algorithme PAES (Pareto-Archived Evolution Strategy) qui s'inspire de l'algorithme (μ+λ)-ES (stratégie d'évolution-(μ+λ)) où μ et λ sont respectivement le nombre de parents et d'enfants. Il inclut une population auxiliaire appelée archive. Pour promouvoir la diversité dans la population, le PAES utilise une technique basée sur un découpage en grilles de l'espace des fonctions objectif. Cette méthode est efficace quand la distribution des solutions dans l'espace de recherche n'est pas uniforme. L'efficacité de la méthode dépend cependant du choix du paramètre de discrétisation de l'espace des fonctions objectif.

Le NSGA-II de Deb et al. (Deb et al., 2000a) est une version améliorée du NSGA développé antérieurement et qui n'utilisait pas le concept d'archive. Ainsi, le NSGA-II maintient les solutions non dominées dans une archive pour les utiliser dans les futures générations. Cet algorithme applique la méthode du surpeuplement pour promouvoir la diversité dans la population. Il permet ainsi de réduire la complexité de l'algorithme, de créer une méthode élitiste et de supprimer l'utilisation de la fonction de partage.

Il existe bien d'autres algorithmes développés par la suite avec pour souci d'améliorer d'avantage les algorithmes évolutifs existants. Il s'agit par exemple de l'algorithme PESA (Pareto Envelopebased Selection Algorithm) de Corne et al. (Corne et al., 2000) . Il reprend le principe de division en grilles utilisé dans le PAES et définit un paramètre appelé facteur de compression ("*squeeze factor*").Le *squeeze factor*

représente la mesure du surpeuplement d'une zone de l'espace. Le PESA est très élitiste et performe moins bien avec les problèmes discrets ou présentant plusieurs optima. Cet algorithme ne garantit pas non plus que les solutions Pareto optimales disposées aux extrémités du front de Pareto soient gardées dans l'archive à chaque génération. Bien qu'il ait une vitesse de convergence élevée, le PESA convient seulement aux problèmes continus et simples.

Le NSGA-II avec contrôle de l'élitisme de (Deb et Goel , 2001) est une autre version du NSGA-II qui vise à assurer une diversité de la population dans tout l'espace faisable. Zitzler et al. (Zitzler et al., 2000) ont également proposé une version révisée du SPEA, qu'ils ont nommé le SPEA2. Cette version calcule la densité de la population autour d'un individu dans l'espace des solutions et l'utilise pour corriger l'expression de sa fonction d'adaptation. Pour assurer la diversité de la population sur le front de Pareto, un opérateur de troncature est utilisé et permet de préserver les solutions aux extrémités du front de Pareto. Le PESA-II de Corne et al. (Corne et al., 2001) adopte une sélection basée sur les zones de l'espace occupées par les individus contrairement aux autres algorithmes qui utilisent la fonction d'adaptation des individus. Après avoir sélectionné une zone, un individu de celle-ci est choisi aléatoirement. Cette méthode répartit efficacement les solutions sur la frontière de Pareto à cause de sa capacité à choisir, avec une plus grande probabilité que le tournoi classique, les individus situés dans des zones moins denses. Bien que cette technique ait permis de faire évoluer positivement la sélection de manière à privilégier les zones de l'espace les moins surpeuplées, elle dépend fortement du réglage du paramètre de discrétisation de l'espace de recherche.

Les algorithmes évolutifs multi-objectifs ont connus une application intense dans le domaine de l'ingénierie pour aider à concevoir des systèmes opérant dans des conditions optimales. Dans la pratique, l'optimisation des problèmes d'ingénierie rencontre diverses difficultés liées notamment aux multiples contraintes dont il faut tenir compte. À cause de ces contraintes, l'espace des solutions faisables se trouve être réduit, puisque plusieurs solutions peuvent ne pas satisfaire aux contraintes du problème et deviennent par conséquent infaisables, c'est-à-dire irréalisables ou

inadmissibles. Pour des problèmes très contraints, il est parfois difficile pour les algorithmes d'optimisation de trouver une seule solution faisable au début de la recherche. Il devient alors important de tenir compte des solutions infaisables durant le processus de l'optimisation. Divers mécanismes sont proposés dans la littérature dont la "*fonction pénalité*", pour aider à intégrer les solutions infaisables dans le processus d'optimisation. Le principe de la *fonction pénalité* est simple et consiste à pénaliser les solutions infaisables en fonction de leur degré de violation des contraintes. Ainsi, la fonction d'adaptation (ou fonction objectif) d'un individu est pondérée par un coefficient de pénalité qui permet de dégrader sa valeur.

Jiménez et al. (Jiminéz et al., 1999), ont proposé la méthode de sélection par tournoi contraint pour gérer les solutions infaisables, mais cette méthode n'a pas de mécanisme pour éviter la convergence prématurée. La méthode proposée par Deb (Deb et al., 2000a) utilise les mêmes règles de la sélection par tournoi contraint, mais prévoit d'utiliser des niches pour préserver la diversité. Cette approche privilégie cependant les solutions faisables au détriment des solutions infaisables, ce qui contredit l'idée générale de garder des solutions infaisables dans la population. Elle présente ainsi des difficultés à trouver l'optimum global si celui-ci se trouve à la frontière entre les régions faisable et infaisable. Coello et al., (Coello et al., 2002) ont également utilisé la sélection par tournoi restreint basée sur les règles de faisabilité et d'une population échantillon comme dans l'algorithme NPGA. Leur algorithme utilise un paramètre pour le contrôle de la diversité dans la population. Cet algorithme présente des difficultés lorsque vient le temps de résoudre les problèmes de grande dimensionnalité. Hernández et al. (Hernandez et al., 2003) ont proposé l'algorithme IS-PAES (Inverted Shrinkable PAES) qui concentre la recherche sur des zones spécifiques de la région faisable en rétrécissant au fur et à mesure l'espace de recherche. L'inconvénient majeur de cet algorithme est le risque de convergence vers un optimum local à cause de la forte pression de sélection.

L'algorithme SMES (Simple Multimembered Evolution Strategy) proposé par Efrén et Coello (2005) utilise le mécanisme auto-adaptatif des stratégies d'évolution multi-membre ((μ+l)-Evolution Strategy) pour prospecter suffisamment l'espace de solutions

afin d'atteindre la région faisable. Son inconvénient majeur provient de l'utilisation de nouveaux paramètres qui ne conviennent pas nécessairement à tous les types de problèmes.

Le développement des algorithmes évolutifs multi-objectifs suscite de nos jours un grand intérêt et différents travaux ont été présentés dans la littérature qui propose des algorithmes d'optimisation généralement adaptés pour certains types de problèmes. Leur utilisation pour traiter d'autres problèmes peut donner des résultats non satisfaisants, ou peut requérir des modifications majeures qui ne sont pas toujours à la portée de l'utilisateur. Le lecteur intéressé pourra trouver plus d'informations sur divers sites internet3 dédiés aux algorithmes évolutifs.

Les travaux comparant les algorithmes évolutionnaires multi objectifs ne laissent pas de doute sur le fait que l'introduction de l'élitisme a mené à des performances nettement supérieures. Par contre, les comparaisons des dernières méthodes élitistes entre elles ne permettent pas de faire un choix sans hésitation. Il est difficile de dire que la performance d'un des algorithmes SPEA2, NSGA-II, et PESA est meilleure de celles de deux autres. . Zitzler et al ont établi qu'aux premières générations, PESA s'approche plus vite de la surface des compromis, mais la performance finale de SPEA2 et de NSGA-II semble être meilleure sur certains problèmes tests au sens de la métrique utilisée dans leur étude. Il ne faut pas oublier que toute analyse comparative des algorithmes évolutionnaires multi objectifs est limitée non seulement par le choix des tests mais aussi des métriques de performance. Nous donnerons ainsi dans la section qui suive une description plus détaillée du NSGA-II qui servira de base pour l'optimisation des paramètres du réseau Ad Hoc proposé dans le cadre de cette thèse.

4.4 Nondominated Sorting Genetic Algorithm II (NSGA-II)

L'algorithme NSGA-II ("Non-dominated Sorting Genetic Algorithm") présenté par Deb et al (Deb et al., 2002) est un des algorithmes évolutifs les plus cités dans la littérature. Il est largement utilisé par plusieurs auteurs pour servir de base de comparaison avec leurs propres algorithmes.

4.4.1 Classification des individus

Dans un premier temps, avant de procéder à la sélection, on affecte à chaque individu de la population un rang « rank » (en utilisant le rang de Pareto). Tous les individus non dominés, de même rang sont classés dans une catégorie, à laquelle on affecte une efficacité qui est inversement proportionnelle au rang de Pareto de la catégorie considérée. Le pseudo code de cette procédure de classification est représenté dans l'algorithme ci dessous. La figure suivante présente un exemple de classification en fronts de Pareto.

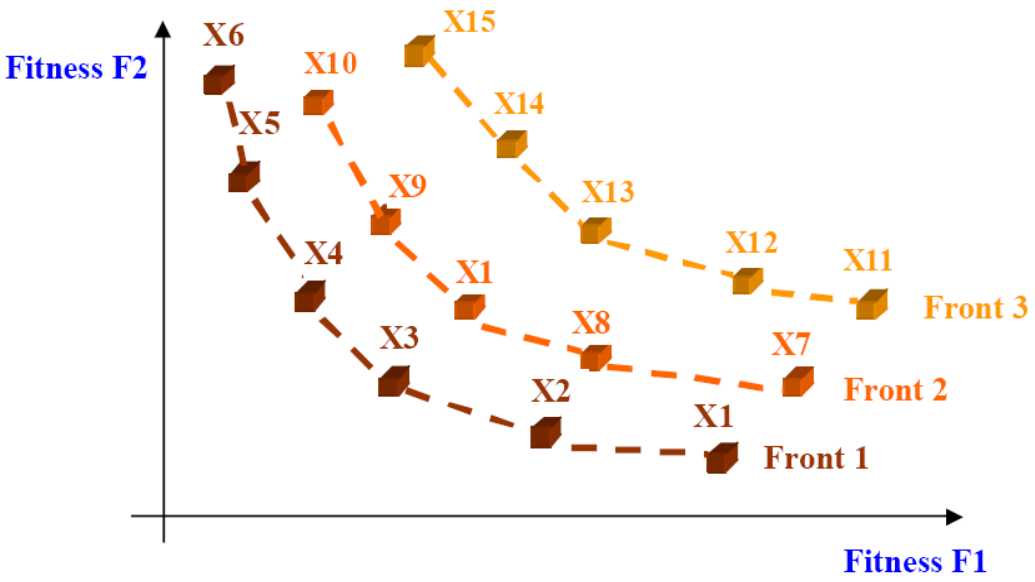

Figure 4.2 Classification des individus suivant le rang de Pareto.

Cet algorithme permet de classer la population globale (parents et enfants) en plusieurs fronts de Pareto. A partir de la population globale P, on sélectionne tous les individus non dominés pour former le front de Pareto d'ordre 2, les individus sélectionnés sont supprimés de la population P. On reprend la même procédure pour former le front de Pareto d'ordre 2 et ainsi de suite jusqu'à P = $\emptyset$.

Algorithme :

```
Début
    P = population()            /*Ensemble total de la population*/
    i = 1                       /*i est le compteur des fronts de Pareto, initialisé à 1*/
    Répéter jusqu'à P = ∅
        Fi = chercher-ind-non-dominés(P)   /*Trouver Fi, l'ensemble des individus
                                non dominés correspondants au front de Pareto i*/
        P = P \ Fi              /*Supprimer les ind. Non dominés à partir de la
                                population globale P*/
        i = i + 1               /*Incrémenter le compteur de front*/
    Fin Répéter
Fin
```

Dans cet algorithme, pour trouver le premier front de Pareto avec une population de N individus, il nous faut N2 comparaison. D'autre part, chaque comparaison est appliquée à M fonctions. Dans ce cas la complexité maximale de l'algorithme est de MN2. On veut maintenant que les individus dans la catégorie considérée se répartissent de manière uniforme au sein de celle-ci. Ou encore on veut qu'il y ait une diversité de solutions.

4.4.2 Préservation de la diversité

Contrairement à l'algorithme NSGA-II, les méthodes évolutionnaires multi-objectifs, ne permettent pas, dans certains cas, d'obtenir une diversité dans la représentation des solutions.

L'algorithme qui suit, montre la technique (***crowding distance***) utilisée par le NSGA-II pour préserver la diversité des solutions sur le front de Pareto. Cette procédure s'applique sur le dernier front de Pareto pour compléter la taille de la population parents pour la génération suivante.

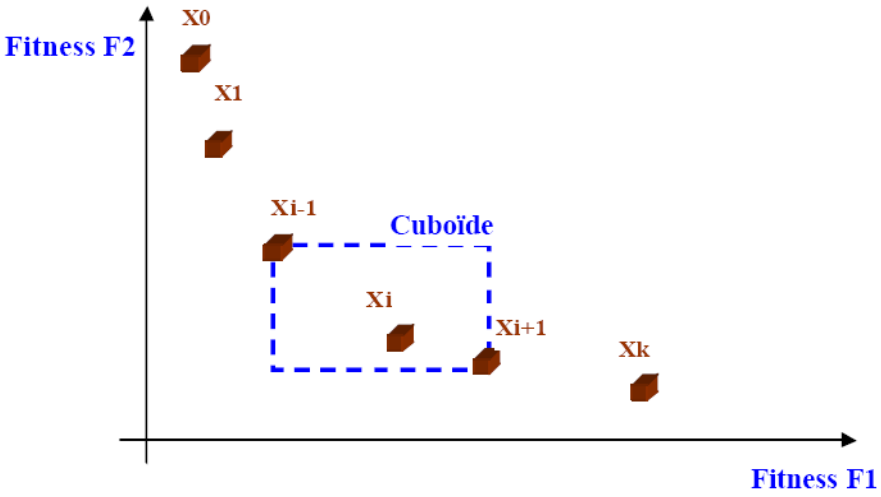

Figure 4.3 Diversité dans la représentation des solutions.

Algorithme

Début

N <- |I| /*Nbr. Des solutions dans le front de Pareto I*/

Pour i = 1 jusqu'à N

I[i]distance <- 0 /*initialisation de la distance*/

Fin pour

Pour m = 1 jusqu'à M /*M est le nombre des fitness*/

I <- sort(I,m) /*tri des individus suivant fitness m*/

I[1]distance <- ◎□# /*pour les deux points extrêmes*/

I[N]distance <- ◎

Pour i = 2 jusqu'à (N – 1) /*pour tous les autres points*/

I[i]distance <- I[i]distance + (I[i+1].m – I[i-1].m)

Fin pour

Fin pour

Fin

où I[i].m représente la valeur de la même fonction objectif relative au i ème individu dans l'ensemble I. la complexité de cette procédure dépend de la répartition de la population.

Quand tous les individus N de la population sont dans le même front I et pour M comparaison (M triages suivant les différentes fonctions objectifs) il nous faut **O(MNlog N)** opération.

4.4.3 Opérateur de comparaison Crowded

L'opérateur de comparaison ***crowded*** ($\prec_n$) guide le processus de la sélection avec la répartition uniforme des solutions de Pareto. Un individu de la population a deux attributs :

- rang de non domination (i_{rank}),
- distance crowding ($i_{distance}$).
- Soit les deux individus i et j,

$$i \prec_n j \text{ si } i_{rank} < j_{rank} \text{ ou } (i_{rank} = j_{rank}) \text{ et } (i_{distance} > j_{distance})$$

Avec cette relation pour la comparaison de deux solutions non dominées appartenant à deux fronts de Pareto, on préfère la solution appartenant au front de Pareto d'ordre le plus faible. Sinon, dans le cas où les deux solutions appartenant au même front de Pareto, on choisit la solution qui a la distance ***crowded*** la plus élevée.

4.4.4 Boucle principale de l'algorithme NSGA-II

Cet algorithme représente la boucle principale du NSGA-II. Initialement, il y a la création aléatoire d'une population parent P_0. Chaque individu de cette population est affecté à un front de Pareto adéquat. On applique les opérateurs génétiques (sélection, croisement et mutation) pour générer la population enfant Q_0 de taille N à partir de la population P_0.

L'élitisme est assuré par la comparaison de la population courante P_t avec la population précédente P_{t-1}.

Algorithme

Début

R_t <- P_t U Q_t /*Combinaison des pop. parent et enfant $|R_t| = 2N$*/

F <- fast-nondominated-sort(R_t) /*F = {F_1, F_2,…} ensemble des fronts de Pareto tel que $\sum|F_i| = 2N$ */

P_{t+1} <- ∅ /*initialisation*/

i <- 1

Tant que $|P_{t+1}| + |F_i| \leq N$ /*jusqu'à la taille max de la pop. N*/

Crowding-distance-assignment(F_i) /*calcul de la distance «crowded» dans Front F_i*/

P_{t+1} <- P_{t+1} U F_i /*Reconstruction de la population parent en ajoutant ième front de Pareto*/

i <- i + 1 /*incrémentation vers le front suivant*/

Fin tant que

Sort(F_i, $<_n$) /*Triage des solution F_i dans l'ordre décroissant suivant $<_n$ */

P_{t+1} <- P_{t+1} U F_i[1:(N - $|P_{t+1}|$)] /*Choisir juste les premiers éléments F_i pour compléter la population parent P_{t+1} jusqu'à N individus*/

Q_{t+1} <- make-new-pop(P_{t+1}) /*Application des opérateurs génétique (sélection, croisement, mutation) pour la génération d'une nouvelle pop. Q_{t+1}*/

t <- t + 1 /*génération suivante*/

Fin

La figure 4.4, décrit les différentes étapes d'évolution de l'algorithme NSGA-II. Dans cette procédure, on distingue quatre étapes :

- Génération d'une population enfants Q_t à partir de la population parents P_t par l'application des opérateurs génétique (croisement et mutation). Cette manipulation est suivie par la fusion de deux populations enfants et parents dans une seule population globale R_t ;
- Classement des individus de la population globale R_t en plusieurs fronts de Pareto dans l'ordre croissant (F1, F2, F3, etc.) ;

- Reconstruction de la population parent pour la génération suivante (P_{t+1}) par la sélection d'individus appartenant aux premiers fronts de Pareto (F1, F2 et F3) tels que :

$$|P_{t+1}| = \sum_{k=1}^{j} |F_k| \leq N$$

où :

N: La taille de la population.

j: Nombre de fronts sélectionnés.

|Fk|: Nombre d'individus dans le front de Pareto d'ordre k.

- Application de la technique de ***crowding*** distance sur le front de Pareto F_{j+1} (Etape F4 dans le cas de la figure 4.4) pour compléter la population P_{t+1} à une taille de N.

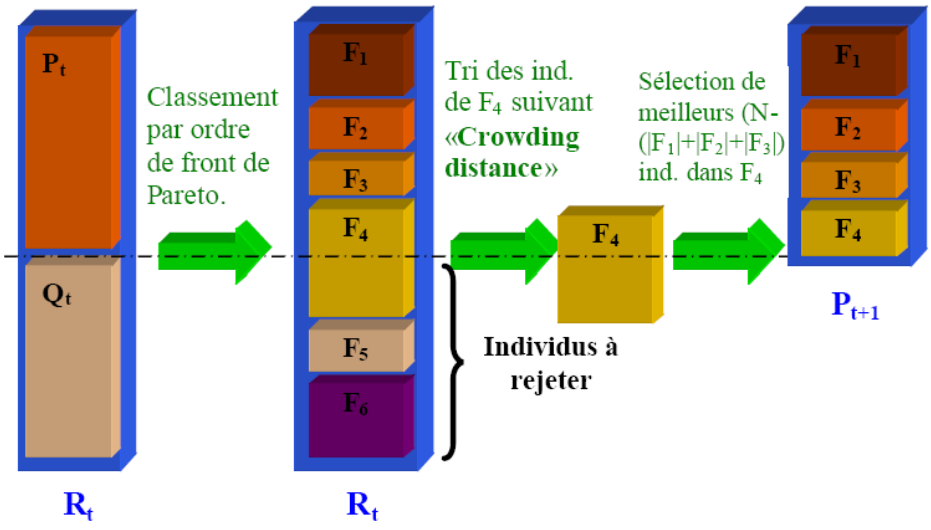

Figure 4.4 Schéma de l'évolution de l'algorithme NSGA –II.

P_t et Q_t dans la figure 5.3, sont respectivement la population parent et enfant de taille N.

« **Crowding distance** » est une mesure de l'encombrement entre les individus de même front de Pareto. La nouvelle population P_{t+1} de taille N est utilisée avec l'application des opérateurs génétiques pour la génération d'une nouvelle population Q_{t+1} de taille N.

4.4.5 Convergence et diversité des solutions des algorithmes multi-objectifs

Au cours de ces dernières années, la recherche sur des algorithmes évolutionnaires a démontré leurs capacités en résolvant les problèmes d'optimisation multi-objectifs, où le but est de trouver, en une seule exécution de l'algorithme, un certain nombre de solutions proches de l'optimale de Pareto. Beaucoup d'études ont été menées sur différent algorithmes évolutionnaires pour montrer leurs aptitudes à progresser les meilleures solutions, avec une bonne diversité (distribution homogène des solutions sur le front de Pareto) des solutions.

Cependant, aucun des algorithmes évolutionnaires multi objectifs a une preuve formelle de convergence avec une large diversité, aux véritables solutions en optimales de Pareto. D'après des études qui ont étés menées sur plusieurs algorithmes multi objectifs, le NSGA-II (Deb et al., 2002) est classé parmi les meilleurs et le plus populaire algorithme jusqu'à ce jour en terme de convergence et de diversité de solutions.

4.5 Conclusion

Ce chapitre sert de complément au chapitre 3, dans lequel nous avons traité le problème de l'optimisation mono-objectif, il expose le problème multi-objectif en donnant les différents algorithmes proposés pour le solutionner tout en mettant l'accent sur l'algorithme NSGA-II qui a été largement détaillé et qui sera utilisé dans chapitre suivant.

Chapitre

5 - Simulations et Résultats

Sommaire

Résumé

Ce chapitre donne l'implémentation d'un algorithme génétique multi-objectifs NSGA-II pour l'optimisation des paramètres du protocole OLSR. Il présente également les expériences et simulations menées pour démontrer de son efficacité.

5.1 Introduction

Dans ce qui a précédé, nous avons vu qu'un réseau Ad Hoc est un ensemble de nœuds mobiles qui sont dynamiquement et arbitrairement éparpillés d'une manière où les interconnexions entre les noeuds peuvent changer à tout moment. Le fait que les noeuds mobiles ne sont pas contrôlés par une seule entité implique que leurs mouvements sont très difficilement prévisibles et que, par conséquent, la connectivité radio change au cours du temps. On est donc en présence de réseaux dont la topologie est dynamique et dont les noeuds ont des caractéristiques particulières (ressources d'énergie et de calcul limitées). Ces spécificités des réseaux Ad Hoc font que les solutions de routage mises au point pour les réseaux filaires classiques ne sont plus adaptées.

Pour répondre à ces contraintes du réseau Ad Hoc nous avons développé un protocole de routage qui associe les bienfaits d'un protocole OLSR et d'un algorithme génétique multi-objectif qui sert comme optimisateur des paramètres du protocole OLSR afin de le rendre adaptatif aux changements de l'environnement (adaptatif à la mobilité des noeuds), chose qui était non seulement manuelle et délicate du fait de son importance et de son difficulté pour des utilisateurs non aguéris et qui devient de ce fait automatique ce qui est un apport non-négligeable.

Dans ce qui suit, nous allons dans premier temps détailler le protocole OLSR utilisé, puis définir les paramètres qui seront optimisés par l'algorithme génétique multi-objectif NSGA-II introduit dans le chapitre précédent. Pour la partie expérimentale, nous présenterons deux séries d'expériences issues du même scénario: 1) avec un nombre de nœuds égale à 20 et une vitesse égale à 10 m/s et 2) le nombre de neouds est égale toujours à 20 et la vitesse égale à 55 m/s, cette dernière expérience sert à simuler la mobilité des nœuds (changement de vitesse de 10 m/s à 55 m/s).

5.2 Le protocole OLSR

5.2.1 Présentation du protocole

Optimized Link State Routing (OLSR) est un protocole par état de lien qui utilise une technique optimisée pour la diffusion des messages topologiques.

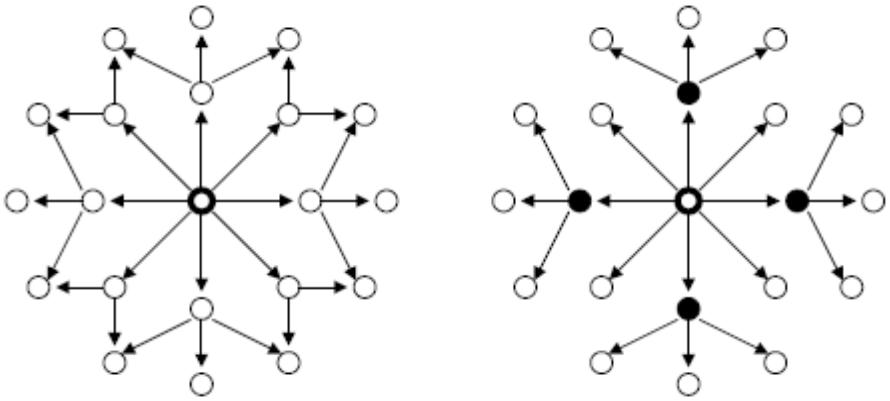

Figure 5.1 Diffusion par inondation (à gauche) et diffusion optimisée (à droite).

La solution consiste a ne permettre qu'à un sous-ensemble de voisins de retransmettre les messages (voir FIG. 5.1). Ces voisins sont appelés les *relais multipoint* ou *MPR*s (MultiPoint Relays). Chaque noeud effectue la selection de ses MPRs en se basant sur la connaissance de son voisinage à deux sauts. L'ensemble des MPRs doit être le plus petit possible, tout en assurant que la diffusion par leur intermédiaire permet d'atteindre le voisinage à deux sauts dans sa totalité. Le problème qui consiste à trouver le plus petit ensemble de MPRs est analogue au problème de la recherche d'ensemble dominant minimal dans un graphe, qui est connu pour être NP-complet. Dans OLSR, les noeuds appliquent une heuristique qui permet de se rapprocher de l'ensemble minimal dans la majeure partie des cas.

5.2.2 Découverte de voisinage

Chaque noeud émet périodiquement des messages appelés "*HELLO*" qui contiennent essentiellement la liste des liens connus vers les voisins directs. La fonction de ces messages est multiple. Ils servent tout d'abord à détecter les voisins directs et la qualité des liens vers ceux-ci, à savoir s'ils sont symétriques ou asymétriques. Comme chaque noeud y publie la liste de ses voisins, il est possible pour un noeud d'acquérir des informations sur son voisinage à deux sauts. Par ailleurs, une fois qu'un noeud a effectué la sélection de ses MPRs, il indique dans ses messages HELLO lesquels de ses voisins sont ses MPRs. Ceci permet à un noeud de savoir quels voisins l'ont choisi comme MPR, autrement dit de constituer son ensemble de *MPR-selectors*.

5.2.3 Diffusion optimisée

Lorsque qu'un noeud reçoit un message de contrôle OLSR, il le traite et ne le transmet que si l'émetteur du message (l'adresse source du message, qui peut être différente de l'adresse de l'émetteur si le message a été généré par un noeud distant) appartient à l'ensemble des MPR-selectors. Cette technique permet de diffuser des messages dans tout le réseau en évitant la saturation.

5.2.4 Les messages topologiques

Les messages topologiques, appelés TC (Topology Control) ne sont émis par un noeud qu'à condition que son ensemble de MPR-selectors n'est pas vide, c'est-à-dire qu'il est MPR d'un de ses voisins. Les messages contiennent la liste des MPR-selectors du noeud. Les noeuds du réseau reconstituent donc une topologie globale mais partielle, puisque tous les noeuds atteignables sont connus, mais pas tous les liens. Cette topologie partielle est néanmoins suffisante pour calculer des chemins minimaux en nombre de sauts vers toute destination.

Le fait de ne permettre qu'aux MPRs de générer des messages TC permet de limiter la quantité de messages diffusés dans le réseau et le fait de ne diffuser que la liste des MPR-selectors permet de limiter la taille des messages.

5.2.5 Les changements topologiques

À chaque changement de topologie, le calcul des routes vers toutes les destinations est déclenché pour mettre à jour les tables de routage. Par ailleurs, lorsque son ensemble de voisins directs ou à deux sauts change, un noeud doit effectuer la sélection de ses MPRs à nouveau.

L'OLSR est un protocole puissant et optimisé. D'une part, il permet de s'adapter parfaitement à la topologie d'un réseau en autorisant chaque équipement à la connaître "localement" et à chaque instant. D'autre part, il est optimisé pour la diffusion car il économise une grande partie de la bande passante du réseau, ce qui est important dans les réseaux denses.

Pour conclure, nous pouvons résumer le fonctionnement du protocole OLSR en trois étapes distinctes:

- Diffusion des messages HELLO pour découvrir le voisinage ;
- Sélection des MPRs : Seuls les relais multipoints relaient l'information ;
- Etablissement et mises à jour des tables de routages via les messages TC.

Ces trois étapes sont modélisés par les paramètres du protocole OLSR à savoir les intervalles HELLO_INTERVAL, TC_INTERVAL et REFRESH_INTERVAL, ces derniers qui seront optimisés par l'algorithme NSGA-II.

5.3 Paramétrage du NSGA-II pour l'optimisation des paramètres OLSR

5.3.1 Définition des variables de décision

Comme mentionné précédemment, le protocole OLSR utilise trois timers pour calculer les trois paramètres (ou intervalles):

- **HELLO_INTERVAL:** Intervalle d'émission des messages HELLO ;
- **TC_INTERVAL:** Intervalle d'émission des messages topologiques TC ;
- **REFRESH_INTERVAL:** Intervalle pour la validité d'un lien entre deux nœuds.

5.3.2 Représentation chromosomique: codage d'individus

Le codage des différents paramètres de l'OLSR est représenté par un chromosome réel. Cette représentation nous permet de coder 3 paramètres soit un espace d'exploration tridimensionnel. Chaque intervalle est codé par une valeur réelle bornée par un min et un max (dans notre cas, min = 1s et max = 30s avec une précision au microsecondes).

Figure 5.2 Représentation des paramètres OLSR.

5.3.3 Fonctions d'objectifs

5.3.3.1 Taux paquets perdus

C'est un élément crucial pour l'évaluation d'un protocole de routage. Afin de déterminer si un protocole est performant, il est utile de savoir s'il minimise au maximise la perte des paquets quelque soit la condition à la qu'elle il est confronté.

Nombre de paquets perdus = calculé depuis le fichier trace.

5.3.3.2 *Délai moyen de transfert*

C'est le délai moyen pris par un paquet de données pour transiter d'une application à une autre.

$$\textbf{Délai moyen de transfert} = \frac{\sum_i (\textbf{tps de la réception du paquet } i - \textbf{tps d'émissionde paquets } i)}{\textbf{nombre de paquets}}$$

5.4 Les outils utilisés

Le développement software peut être divisé en trois parties :

1. Développement sous C++ de l'algorithme génétique NSGA-II ;

2. Combinaison du protocole OLSR développé par l'université de Murcia[1] et du NSGA-II développé ;

3. Intégration du nouveau protocole sous NS2 et simulations, le tout tourne sous linux Ubuntu.

- **Ns2 :** NS-2[2] est un logiciel de simulation de réseaux informatique développé lors d'un projet de la DARPA. Le simulateur se compose d'une interface de programmation en tcl et d'un noyau écrit en C++ dans lequel la plupart des protocoles réseaux ont été implémentés :

  * Couche MAC CSMA, CDMA, 802,X, Token ring, MPLS, liens satéllite ;

  * Couche Réseaux IP, routage dans les réseaux Ad Hoc (aodv, dsr , dsdv, tora, amodv), routage dans les réseaux filaire (Link state, Distance vector), les réseaux multicast, IntServ, DiffServ ;

  * Couche Transport TCP, UDP;

[1] http://masimum.dif.um.es

[2] http://www.isi.edu/nsnam/ns/

* Traffic parreto, ON/OFF, CBR, FTP, telnet

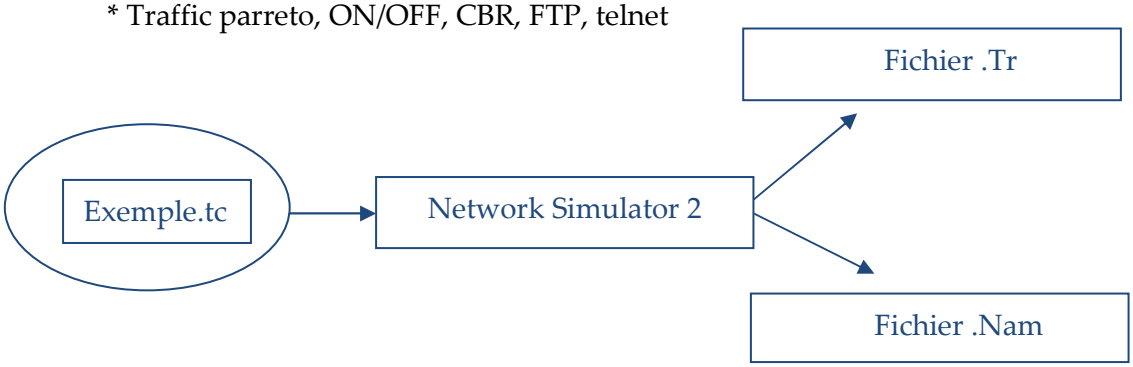

Figure 5.3 Processus de simulation avec NS2.

Ns2 utilise comme entrée un fichier en langage tcl, ce fichier contient le scenario du réseau qu'on veut simuler. Apres la simulation, Ns va génère deux fichier: fichier trace et fichier Nam.

- **Fichier Trace:** un avec une extension tr qui contient tous les événements réalise dans le scenario. Ce fichier est généralement utilisé pour calculer les paramètres liés au performance du réseau par exemple le taux des paquets perdus, le temps moyen des paquets …etc ;
- **Fichier Nam:** Ce fichier est utilisé avec un programme appelé animateur pour visualisé le scenario décrit dans le ficher d'entrée tcl.

5.5 Déroulement de l'expérience

Le déroulement de l'expérience s'effectue comme suit :

1. Génération de la population initiale (ensemble de chromosomes contenant les triplets des intervalles : HELLO_INTERVAL, TC_INTERVAL et REFRESH_INTERVAL) ;

2. Evaluation de chaque chromosome en simulant le protocole OLSR sous NS2 avec le triplet des intervalles ;

3. Si les critères d'arrêts sont vérifiés (nombre de paquets perdus et min et le délai moyen de transfert des paquets est min), l'algorithme génétique s'arrête et on

sauvegarde le best triplet. Sinon on applique les opérateurs de l'algorithme génétique pour générer les nouveaux chromosomes, et on reboucle à l'étape 2.

Il est à noter qu'une fois l'algorithme converge, nous obtenons un ensemble de triplets représentant le front de Pareto. Ces triplets ont le même coût global pour les deux objectifs. Il nous incombe à nous choisir le triplet qui minimise l'un des deux objectifs ou bien les deux à la fois.

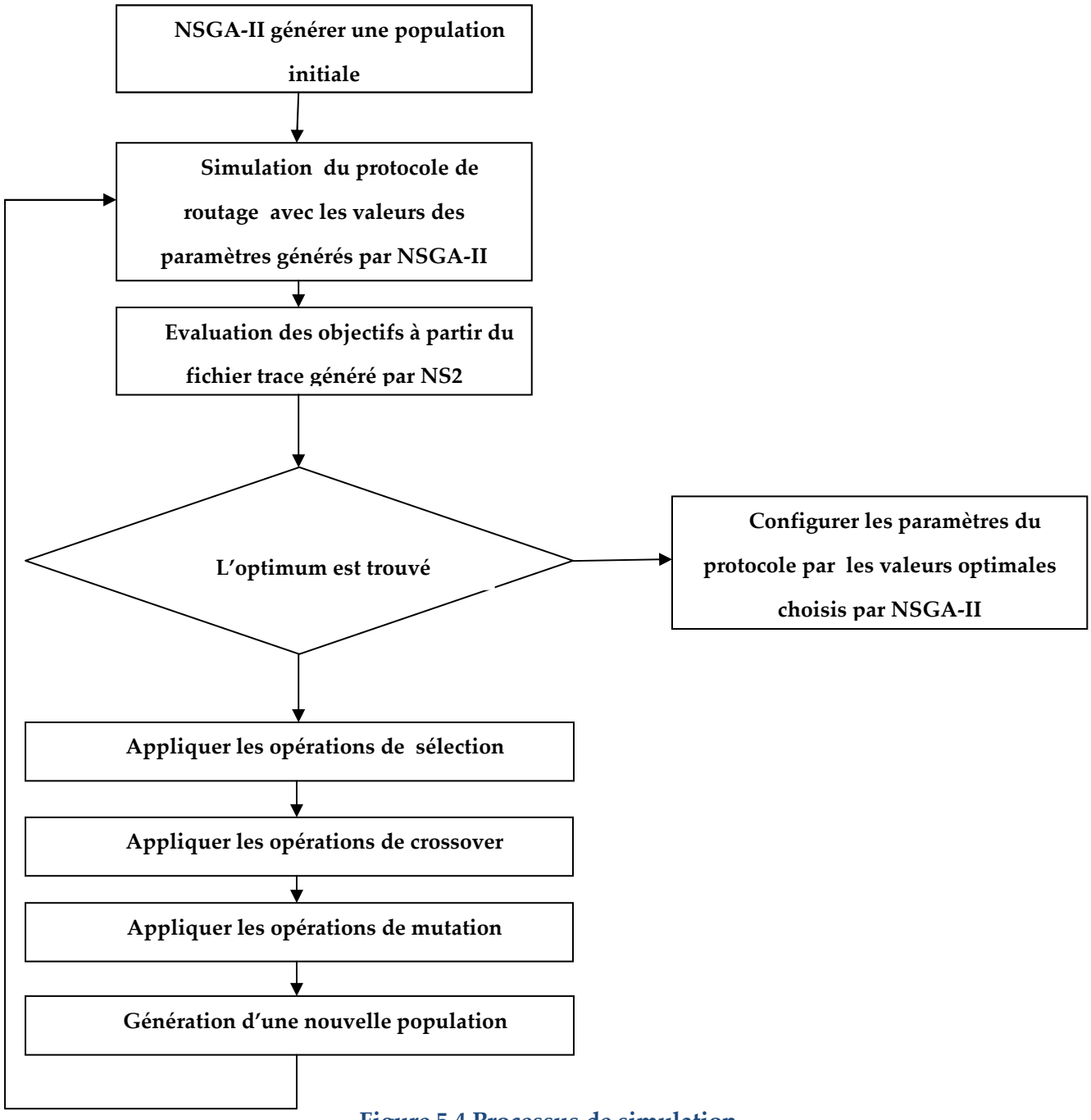

Figure 5.4 Processus de simulation.

5.6 Simulation et résultats

5.6.1 Expérience 1

Cette première expérience illustre l'utilisation de l'algorithme génétique développé (NSGA-II) pour l'optimisation des paramètres du protocole OLSR. Pour cela, nous avons pris les paramètres définis dans le tableau 5.1.

Tableau 5.1 Paramètres de simulation

Paramètres OLSR		Paramètres NSGA-II	
Nombre de nœuds	20	Taille de la Population size	100
Nombre maximum des connections	20	Nombre d'objectifs	2
Vitesse minimale des nœuds	5 m/s	Variables de décisions	3
Vitesse maximale des nœuds	10 m/s	Limite Min (pour les 3 variables)	1
Temps de Pause	5 s	Limite Max (pour les 3 variables)	30
Longueur de la zone de simulation	1000 m	Probabilité du crossover	0.6
Largeur de la zone de simulation	1000 m	Probabilité de mutation	0.1

Le tableau 5.2 donne un extrait du fichier généré par l'algorithme NSGA-II (fichier pour une population initiale) incluant les objectifs à minimiser ainsi que les paramètres à optimiser.

Tableau 5.2 Extrait du fichier généré par NSGA-II (population initiale)

Objectif 1	Objectif 2	Hello intervalle	TC intervalle	Refresh intervalle	Rang	Distance
9,86E-03	5,93E+02	7,88E+00	7,99E+00	1,88E+01	10	1,00E+14
1,00E+14	0,00E+00	2,68E+01	1,75E+01	4,86E+00	1	1,00E+14
1,00E+14	0,00E+00	1,74E+01	1,88E+01	2,83E+00	1	0,00E+00
1,30E-02	7,73E+02	1,77E+00	1,01E+01	1,16E+01	14	1,00E+14
1,00E+14	0,00E+00	2,11E+01	2,93E+01	5,75E+00	1	0,00E+00
……	…….	…….	…….	…….	….	……..

La figure 5.5 donne la distribution des triplets de paramètres (un point par triplet de paramètres Hello intervalle, TC intervalle et Refresh intervalle) dans l'espace des objectifs (pour la population initiale).

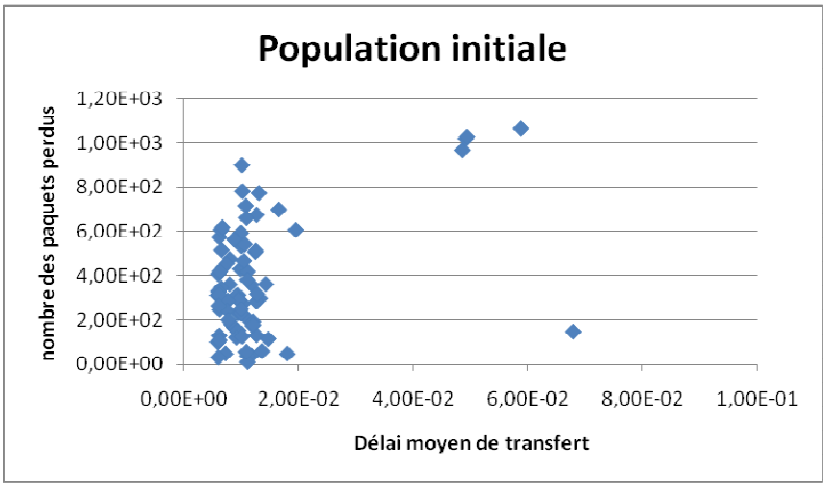

Figure 5.5 Distribution des chromosomes de la population initiale dans l'espace des objectifs.

Le tableau 5.3 (et la figure 5.6) donne les trois points représentants les trois triplets de paramètres trouvés par algorithme NSGA-II. Ces triplets de paramètres représentent trois solutions possibles pour l'initialisation (ou la configuration) du protocole OLSR.

Tableau 5.3 Population finale

Objectif 1	Objectif 2	Hello intervalle	TC intervalle	Refresh intervalle	Rang	Distance
5,75E-03	9,10E+01	2,83E+01	1,06E+01	1,80E+01	1	1,00E+14
5,85E-03	0,00E+00	2,83E+01	2,88E+01	2,29E+01	1	1,00E+14
5,82E-03	1,80E+01	2,68E+01	8,09E+00	1,05E+01	1	4,01E-01

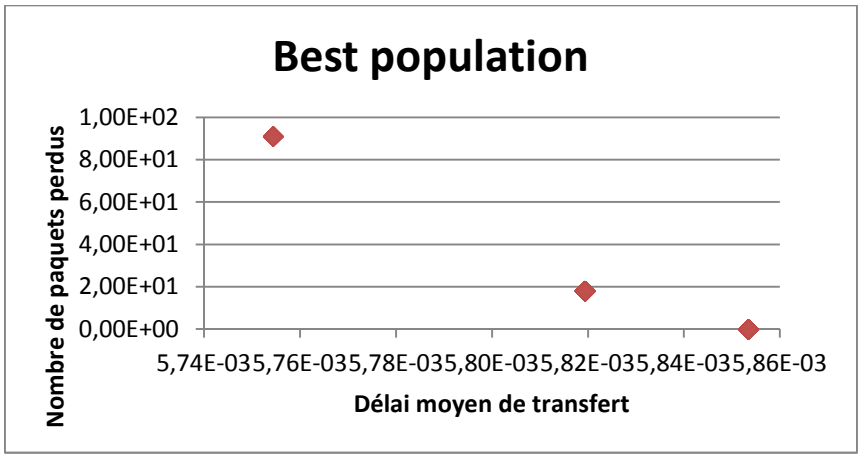

Figure 5.6 Distribution des chromosomes de la population finale dans l'espace des objectifs.

La figure 5.7 donne le front de Pareto trouvé par notre algorithme.

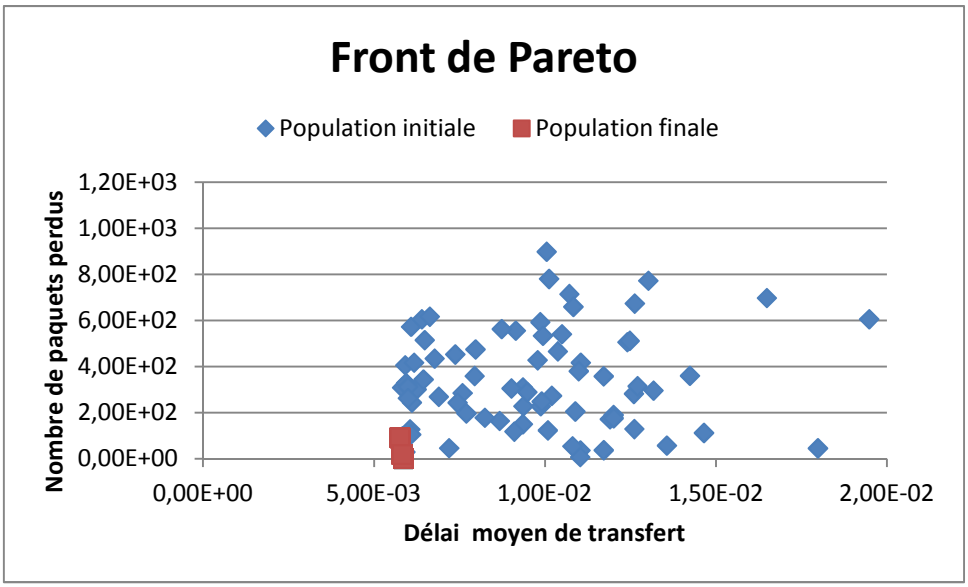

Figure 5.7 Front de Pareto.

Tandis que la figure 5.8 montre bien que nous avons plus que trois triplets possibles (plusieurs points superposés).

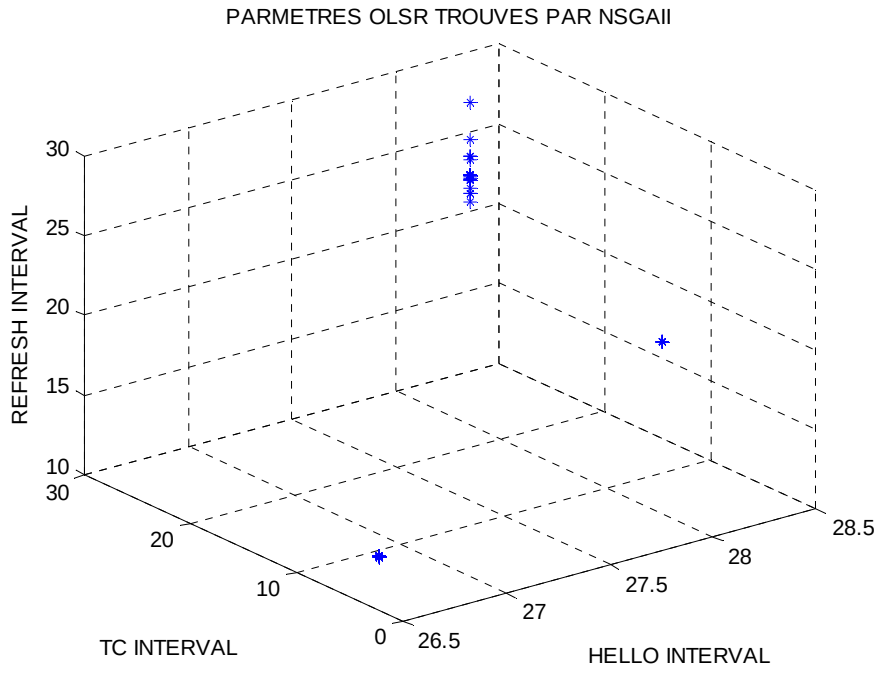

Figure 5.8 Distribution des paramètres OLSR finaux.

5.6.2 Expérience 2

La deuxième expérience sert à simuler la mobilité des nœuds en variant la vitesse de déplacement. Le tableau 5.4 donne les nouveaux paramètres de simulation.

Tableau 5.4 Paramètres de simulation

Paramètres OLSR		Paramètres NSGA-II	
Nombre de nœuds	20	Taille de la Population size	100
Nombre maximum des connections	20	Nombre d'objectifs	2
Vitesse minimale des nœuds	5 m/s	Variables de décisions	3
Vitesse maximale des nœuds	55 m/s	Limite Min (pour les 3 variables)	1
Temps de Pause	5 s	Limite Max (pour les 3 variables)	30
Longueur de la zone de simulation	1000 m	Probabilité du crossover	0.6
Largeur de la zone de simulation	1000 m	Probabilité de mutation	0.1

Le tableau 5.5 donne un extrait du fichier généré par l'algorithme NSGA-II (fichier pour une population initiale) incluant les objectifs à minimiser ainsi que les paramètres à optimiser.

Tableau 5.5 Extrait du fichier généré par NSGA-II (population initiale)

Objectif 1	Objectif 2	Hello intervalle	TC intervalle	Refresh intervalle	Rang	Distance
9,73E-03	5,26E+02	7,88E+00	7,99E+00	1,88E+01	18	1,00E+14
1,32E-02	4,32E+02	1,77E+00	1,01E+01	1,16E+01	23	2,78E-01
1,00E-02	1,29E+02	1,60E+01	2,40E+01	2,96E+01	17	1,00E+14
6,61E-03	0,00E+00	2,75E+01	1,55E+01	1,79E+01	8	1,00E+14
1,03E-01	2,88E+02	6,69E+00	1,97E+01	2,89E+01	21	1,00E+14
.....	.....	.....	.....	.....	....	.....

La figure 5.9 donne la distribution des triplets de paramètres (un point par triplet de paramètres Hello intervalle, TC intervalle et Refresh intervalle) dans l'espace des objectifs (pour la population initiale).

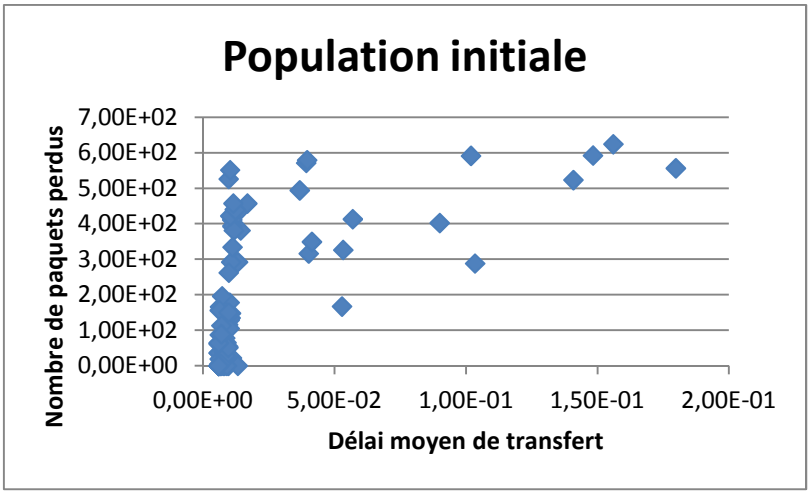

Figure 5.9 Distribution des chromosomes de la population initiale dans l'espace des objectifs.

Le tableau 5.6 (et la figure 5.10) donne un point représentants les trois triplets de paramètres trouvés par algorithme NSGA-II. Ces triplets de paramètres représentent trois solutions possibles pour l'initialisation (ou la configuration) du protocole OLSR.

Tableau 5.6 Population finale

Objectif 1	Objectif 2	Hello intervalle	TC intervalle	Refresh intervalle	Rang	Distance
5,79E-03	0,00E+00	2,42E+01	2,47E+01	1,04E+01	1	1,00E+14
5,79E-03	0,00E+00	2,42E+01	2,47E+01	9,83E+00	1	1,00E+14
5,79E-03	0,00E+00	2,42E+01	2,47E+01	9,81E+00	1	0,00E+00

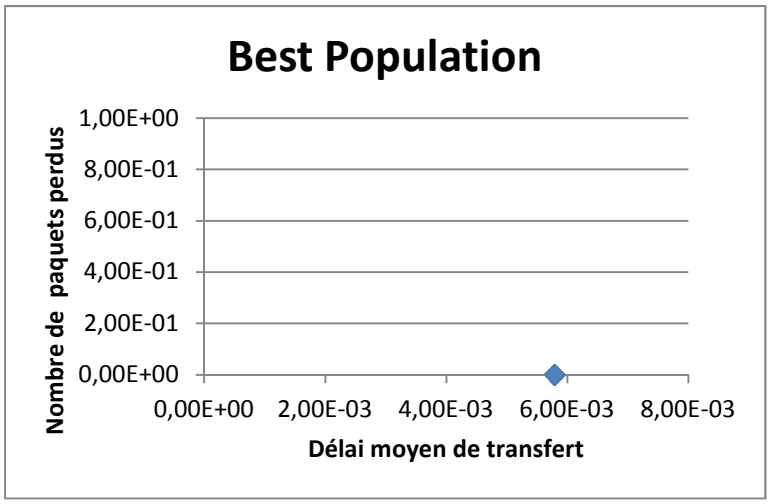

Figure 5.10 Distribution des chromosomes de la population finale dans l'espace des objectifs.

La figure 5.11 donne le front de Pareto trouvé par notre algorithme.

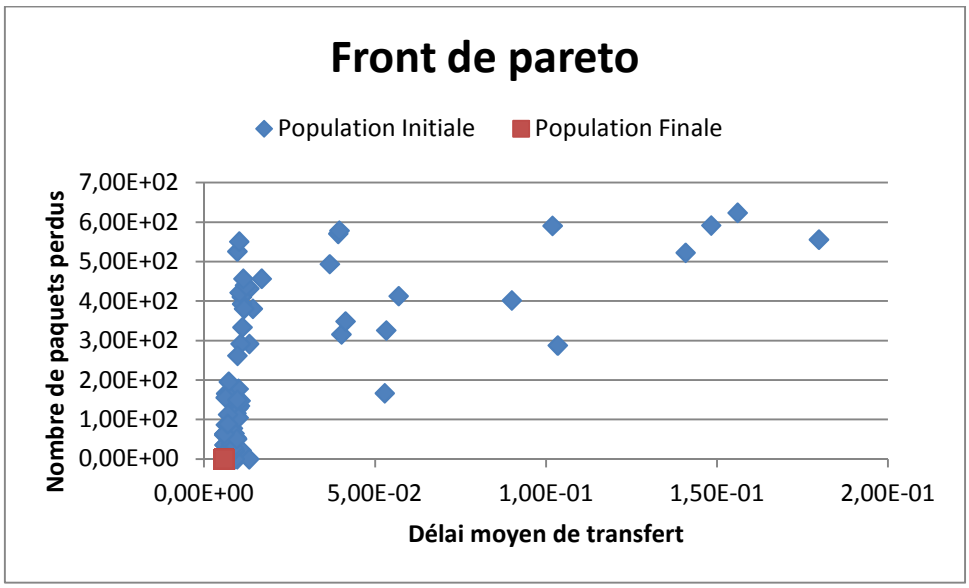

Figure 5.11 Front de Pareto.

Tandis que la figure 5.12 montre bien que nous avons plus que trois triplets possibles (plusieurs points superposés).

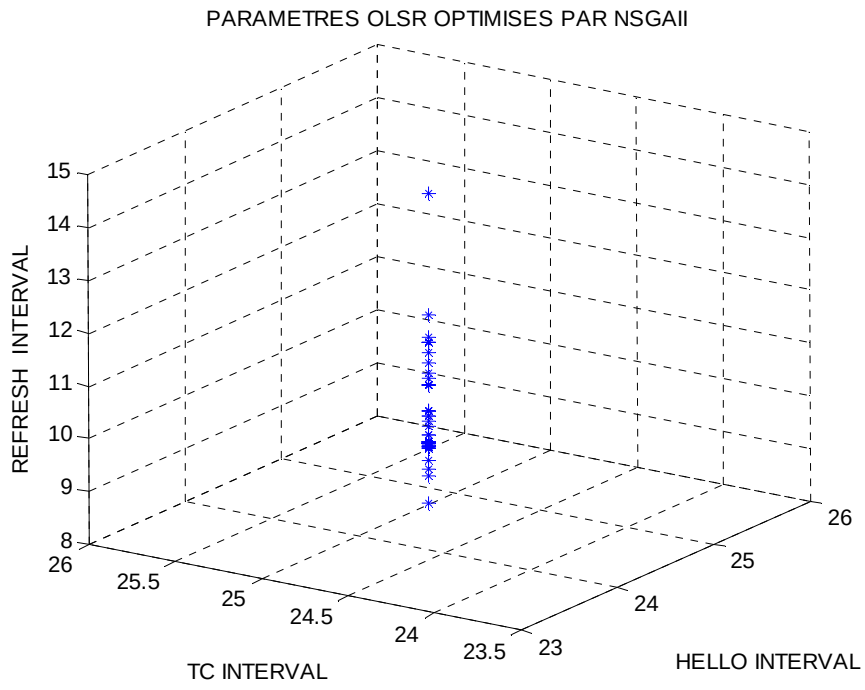

Figure 5.12 Distribution des paramètres OLSR finaux.

5.7 Conclusion

Ce chapitre a montré l'efficacité de l'algorithme génétique développé pour l'automatisation et l'optimisation des paramètres du protocole OLSR. En effet, le choix des paramètres dans un espace multi-paramètres et multi-objectifs est très difficile voire même impossible, car il est difficile de choisir les paramètres en tenant compte de la corrélation qui existent entre ces derniers, chose qui est faite sans soucis par l'algorithme NSGA-II qui fonctionne sur un espace de paramètres (avec une population de solutions) plutôt que sur un paramètre unique à la fois, en plus cette version de l'algorithme génétique travaille bien sur plusieurs objectifs plutôt que sur un seul à la fois ce qui nous a permis d'estimer les bons paramètres tout en minimisant les deux objectifs choisis à priori. En plus, les deux expériences de ce chapitre montrent bien la réaction de l'algorithme génétique pour suivre le changement de l'environnement du protocole OLSR (ex. la mobilité). Cela veut dire que l'algorithme génétique non seulement il permet d'automatiser le choix des paramètres, mais en plus c'est un choix optimal vu qu'il minimise l'ensemble des objectifs et en plus il est adaptatif au changement d'environnement.

Conclusion générale

Le concept des réseaux mobiles Ad Hoc permet d'étendre les notions de la mobilité à toutes les composantes de l'environnement. Le problème de routage, en particulier, a suscité un vif intérêt dans la communauté des chercheurs. Plusieurs travaux de recherche se sont intéressés au problème de routage dans les réseaux Ad Hoc. Bien que les protocoles proposés présentent certaines caractéristiques pertinentes, ils présentent certaines limites, surtout à forte mobilité des noeuds ou à forte charge du réseau. Dans cette thèse nous utiliserons un algorithme génétique afin d'automatiser la sélection des valeurs des paramètres Ad Hoc et donc minimiser les effets d'inondations massives tout en assurant une meilleure adaptation aux changements fréquents de topologie.

Les problèmes d'optimisation requièrent des méthodes robustes, efficaces et flexibles. Afin de s'affranchir de la complexité du problème d'adaptation, de réduire le coût de réalisation et de simplifier le modèle étudié. Nous avons utilisé un algorithme génétique comme outil plus adapté à notre problématique aux contraintes multi-objectives. Nous ne pouvons s'attaquer à cette tache sans donner les bases et les concepts fondamentaux de l'optimisation mono-objectif plus particulièrement dans le cas de l'utilisation d'un algorithme génétique, ce qui nous a conduit à détailler ce dernier pour tout lecteur de ce document, et nous a facilité l'introduction au problème multi-objectif en donnant les différents algorithmes proposés pour le solutionner tout

en mettant l'accent sur l'algorithme NSGA-II qui a été largement détaillé et qui a été utilisé avec succès dans chapitre suivant.

Les résultats de simulation ont montré l'efficacité de l'algorithme génétique développé pour l'automatisation et l'optimisation des paramètres du protocole Ad Hoc. En effet, le choix des paramètres dans un espace multi-paramètres et multi-objectifs est très difficile voire même impossible, a été fait sans soucis par l'algorithme NSGA-II qui fonctionne sur un espace de paramètres (avec une population de solutions) plutôt que sur un paramètre unique à la fois. En plus, le NSGA-II nous a permis d'estimer les bons paramètres tout en minimisant les deux objectifs choisis à priori. Les deux expériences ont montré également la réactivité de l'algorithme génétique à tout changement d''environnement du protocole (ex. la mobilité). Cela veut dire que l'algorithme génétique non seulement il permet d'automatiser le choix des paramètres, mais en plus c'est un choix optimal qui minimise l'ensemble des objectifs en plus d'être adaptatif au tout changement d'environnement.

Ce travail ouvre de nouvelles perspectives d'optimisation des paramètres d'un routage proactif et sa nature adaptative à tout environnement changeant mérite une continuité des travaux pour valider la robustesse de ce dernier ainsi que sa généralisation à d'autres types de protocoles.

Références bibliographiques

(Abolhasan et al., 2003) M. Abolhasan, T. Wysocki & E. Dutkiewicz – « A review of routing protocols for mobile ad hoc networks », *Ad Hoc Networks* **2** (2003), no. 1, pp. 1–22.

(Badache et al., 2002) N. Badache, D. Djenour, A. Derhab & T. Lemlouma – « Les protocoles de routage dans les réseaux mobiles ad hoc », *Revue d'Information Scientifique et Technique (RIST)* **12** (2002), no. 2, p. 77–112.

(Berro, 2001) A. Berro, "Optimisation multi-objectif et stratégie d'évolution en environnement dynamique », Thèse de doctorat, Université des Sciences Sociales, Toulouse I, France, 2001.

(Chlamtac, 2003) I. Chlamtac, M. Conti, and J. Liu. "Mobile Ad Hoc networking: imperatives and challenges". Ad Hoc Networks, pp13–64, 2003.

Clausen et Jacquet, 2008) T. Clausen & P. Jacquet – « Optimized Link State Routing Protocol (OLSR) », RFC 3626, Internet Engineering Task Force (IETF), 2003, Last draft 2008: draft-ietf-manet-olsrv2-07.txt.

(Clausen et al., 2001) T. H. Clausen, G. Hansen, L. Christensen, and G. Behrmann, The optimized link state routing protocol, evaluation through experiments and simulation, IEEE Symposium onWireless Personal Mobile Communication 2001, Septembre 2001.

(Coello et al., 2002) C. A Coello. ,D.V Veldhuizen, G.B Lamont. Evolutionary algorithms for solving multi-objective problems. Kluwer Academic Publishers, Boston. (2002).

(Corne et al., 2001) D.W Corne,N.R Jerram,J.D Knowles,M.J Oates. PESA-II: region-based selection in evolutionary multiobjective optimization. Proceedings of the Genetic and Evolutionary Computation Conference 2001, pp. 283-290.2001.

(Corne et al., 2000) D.W Corne, J.D Knowles, M.J Oates. The Pareto envelope-based selection algorithm for multiobjective optimization. Proceedings of the Parallel Problem solving from Nature VI conference 2000, p. 839-848.

(Corson et Macker, 1999) S. Corson, J. Macker, 'Mobile Ad Hoc Networking (MANET) : Routing Protocol Performance Issues and Evaluation Considerations', Request for Comments 2501, *IETF*, January 1999.

(Darwin, 1859) C. Darwin. The Origin of Species by Means of Natural Selection. Mentor Reprint, 1958, NY, 1859.

(Deb et al., 2002) K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II", KanGAL Report No. 200001, 2002.

(Deb et Goel , 2001) K. Deb ,T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. Proceedings of the first International Conference on Evolutionary Multi-criterion Optimization 2001 , pp. 67-81, Berlin.

(Deb et al., 2000a) K. Deb, S. Agrawal , A. Pratap , T. Meyarivan .A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Parallel Problem Solving from Nature (PPSN VI) 2000, pp. 849-858, Berlin.

(Deb et Agrawal, 1994) K. Deb, R. B. Agrawal . Simulated binary crossover for continuous search space. Complex Systems 1994, vol. 9, pp. 115-148.

(Dorigo al., 1991), M.Dorigo, A. Colorni et V. Maniezzo. Distributed Optimization by Ant Colonies. Proceedings of the European Conference on Artificial Life ECAL'91, p. 134-142. Elsevier Publishing, 1991.

(Elorrieta, 2007) D. Elorrieta. Protocoles de routage pour l'interconnexion des réseaux ad-hoc et UMTS, Mémoire de master, Université Libre de Bruxelles, 2007.

(Fonseca et Fleming, 1993) C.M Fonseca. et P.J Fleming. Genetic Algorithms for Multiobjective Optimization : Formulation, Discussion and Generalization , in Genetic Algorithms : Proceedings of the Fifth International Conference , S. Forrest, editeur, San Mateo, CA : Morgan Kaufmann, juillet 1993.

(Hernandez et al., 2003) A.A Hernandez. S.R Botello.L.G Lizarraga. C.A.C Coello . IS-PAES: A constraint handling technique based on multiobjective optimization concepts. Proceedings of the 2nd Conference on Evolutionary Multiobjective Optimization 2003 , pp. 73-87, Faro, Portugal.

(Ghosekar et al., 2010) P. Ghosekar Mobile, G.Katkar, P.Ghorpade . Ad Hoc Networking: Imperatives and Challenges.2010.

(Goldberg, 1989) D.E Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Reading MA Addison Wesley, 1989.

(Glover, 1986) F. Glover. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research, 5, p. 533-549, 1986.

(Haas, 1997) Z. J. Haas – « A new routing protocol for the reconfigurable wireless networks », dans Proc. 6th IEEE Int'l Conf. on Universal Personal.1997.

(Holland, 1975) J.Holland , adaptation in naturel and artificial systems, mit press, 1975.

(Horn et al., 1994) J.Horn ,N. Nafpliotis ,D.E Goldberg .A niched Pareto genetic algorithm for multiobjective optimization. Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE World Congress on Evolutionary Computation, vol. 1, pp. 82-87, Piscataway, NJ, IEEE Press,1994.

(Jiminéz et al., 1999) F. Jiminéz , J.L Verdegay. Evolutionary techniques for constrained optimization problems. 7th European Congress on Intelligent Techniques and Soft Computing, Aachen, Germany.1999.

(Joa-Ng et Lu, 1999) M. Joa-Ng , I.-T. Lu . A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks , *IEEE Journal on Selected Areas in Communications* **17** (1999), no. 8, p. 1415–1425.

(Johnson et al., 2007) D. A. Johnson, Y.-C. Hu , D. A. Maltz . The Dynamic Source Routing protocol (DSR) for mobile ad hoc networks for ipv4 the Dynamic Source Routing protocol (DSR) for mobile ad hoc networks for ipv4 , RFC 4728, Internet Engineering Task Force (IETF), 2007, rfc4728.txt.

(Johnson et al., 1996) B. Johnson, A. Maltz, and J. Broch, Dynamic source routing in Ad Hoc network, Mobile Computing, Kluwer Academic Publishers, 1996, pp. 153–181.

(Kirpatrick et al., 1983) S.Kirpatrick,& al 83, C.D.Gelatt and M.P.Vecchi, "optimization by simulated annealing", science,220(4598),pp 671-680,june 1983.

(Lemlouma, 2000) T.LEMLOUMA. Le Routage dans les Réseaux Mobiles Ad Hoc, Université Houari Boumediene, Institut d'Informatique, Mini projet, Sep2000.

(Michalewicz, 1992) Z.Michalewicz. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag , (1992).

(Michalewicz et Janikov, 1991) Z.Michalewicz and C.Z. Janikov. Handling constraints in genetic algorithms. In Proceedings of the Fourth International Conference on Genetic Algorithm. ICGA, 1991.

(Minoux, 1983) M. Minoux, Programmation Mathématique : Théories et Algorithmes, Dunod, vol. 1, Paris, 1983.

(Perkins et al., 2003) C. E. Perkins, E. Belding-Royer , S. Das . Ad hoc On-demand Distance Vector (AODV) Routing , RFC 3561, Internet Engineering Task Force (IETF), 2003, rfc3561.txt.

(Perkins et al., 2001) C. E. Perkins, E. E. Royer, S. R. Das, and M. K. Marina, Performance comparison of two on-demand routing protocols for Ad Hoc networks, In IEEE Personal Communications, Feb 2001, vol. 8, pp. 16–28.

(Perkins et Bhagwat, 1994) E. Perkins and P. Bhagwat . Highly dynamic destination-sequenced distance vector routing (dsdv) for mobile computers, SIGCOMM Comput. Commun. Rev. 24 (1994), no. 4, 234–244.

(Royer et Toh, 1999) M.E Royer, C. Toh. *A review of current routing protocols for Ad Hoc mobile wireless networks.* IEEE Personal Communications, pp 46-55, avril 1999.

(Salomo, 2001) M.Salomo, 2001. Etude de la parallésation de méthodes heuristiques d'optimisation combinatoire, thèse de doctorat, université louis pasteur strasbourg 1, décembre.2001.

(Théoleyre, 2006) F. Théoleyre . Une auto-organisation et ses applications pour les réseaux ad hoc et hybrides , Thèse de doctorat, INSA de Lyon, Lyon, France, 2006.

(Zitzler et al., 2000) Zitzler, E., Laumanns, M., Thiele, L. (2001). SPEA2: improving the strength Pareto evolutionary algorithm. Technical report 103, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), Zurich,2001.

الخلاصة

إن مشكلة التوجيه في الشبكات "أد هوك" اعير اهتماما كبيرا في الأوساط البحثية .وعلى الرغم من البروتوكولات المقترحة في البحوث الحالية تتميز بخصائص مميزة إلا أنها تبدو غير كافية ولا سيما في حالة نشاط الشبكة أو في حالة كثافة غير عادية للشبكة. هاته الخصائص تختار عادة من قبل خبير مختص وبطريقة يدوية. تقترح هاته المذكرة طرية جديدة لإختيار هاته الخصائص بطريقة أوتوماتيكية وهذا باستعمال خوارزمية جينية متعددة الأهداف متعددة، حيث مكن هذا الإستعمال من إختيار الخصائص زيادة على تقليل التأخير متوسط نقل حزمة معلومات في الشبكة وعدد الحزم المفقودة كما مكن أيضا من التكيف مع كل تغيير في طبولوجية الشبكة مما يجعل البروتوكول المقترح أوتوماتيكي الخصائص تكيفي.

الكلمات الشائعة: شبكات أد هوك، خوارزمية جينية، بروتوكول.

Résumé

Le problème de routage dans les réseaux Ad Hoc, en particulier le routage proactif, a suscité un vif intérêt dans la communauté des chercheurs. Bien que les protocoles proposés dans la littérature présentent certaines caractéristiques pertinentes, ils présentent certaines limites, surtout à forte mobilité des noeuds ou à forte charge du réseau qui dépendent des paramètres du réseau Ad Hoc choisis souvent d'une manière intuitive par un expert aguerri. Cette thèse détaille nos travaux pour solutionner cette tâche délicate et généralement manuelle utilisant un algorithme génétique pour l'automatisation de la sélection des paramètres. L'utilisation de l'algorithme génétique multi-objectif permet non seulement de minimiser le délai moyen de transfert du réseau et du nombre des paquets perdus mais en plus de s'adapter à chaque changement de topologie ce qui le rend adaptatif à tout environnement.

Mots clés : réseaux Ad Hoc, algorithme génétique, protocole.

Abstract

The problem of routing in Ad Hoc networks, in particular proactive routing, has attracted the attention of researchers. Although the protocols proposed in the literature present some relevant characteristics, they have limitations, especially highly mobile nodes or high network load-dependent parameters of the Ad Hoc network often chosen intuitively by a seasoned expert. This thesis describes our work to solve this difficult task using a genetic algorithm for automatic parameter selection. The use of multi-objective genetic algorithm can not only minimize the average delay of transfer of the network as well as the number of lost packets but also to adapt to each change of topology which makes it adaptive to any environment.

Keywords: Ad Hoc networks, genetic algorithm, protocol.