الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية العلوم

قسم: الإعلام الآلي

Mémoire de Master

Spécialité : Réseaux informatiques et systèmes répartis (RISR)

Thème

Traffic light timing optimization

Présenté par :
CHAFI Sara Sanaa

HIRECHE Meryem

Dirigé par :
Dr. KOUIDRI Siham (Encadreur)
Dr. KOUIDRI Chaima (Co-encadreur)

Promotion 2023 - 2024

ملخص

أنظمة الإشارات الذكية أساسية لتحسين حركة المرور الحضرية. إنها تعدل إشارات المرور في الوقت الفعلي، مما يقلل من الازدحام ويحسن التدفق المروري. يركز بحثنا على تحسين أنظمة إشارات المرور لتعزيز انسيابية حركة المرور الحضرية. بعد دراسة حالة التقدم المحرز في مجالي أنظمة إشارات المرور والخوارزميات المتقدمة، نقترح نهجًا لضبط الإشارات في الوقت الفعلي، مما يقلل من الازدحام واستهلاك الوقود. يهدف هذا النهج إلى حل مشكلة جدولة إشارات المرور صعبة الحل، بهدف تقليل الازدحام وطوابير الانتظار إلى أدنى حد.

الكلمات المفتاحية: إشارات المرور، النقل، الشبكة الطرقية، الخوارزميات المتقدمة، تقليل طوابير الانتظار.

Abstract

Intelligent signaling systems are essential for optimizing urban traffic. They adjust traffic signals in real-time, minimizing congestion and improving traffic flow. Our research focuses on optimizing traffic signal systems to enhance urban traffic fluidity. After studying the state of the art in traffic signal systems and metaheuristics, we propose an approach to adjust signals in real-time, thereby reducing congestion and fuel consumption. This approach aims to solve the NP-hard problem of signal timing scheduling, with the objective of minimizing queues and traffic jams.

Keywords: traffic signals, transportation, road network, metaheuristic, queue minimization.

Résumé

Les systèmes de signalisation intelligents sont essentiels pour optimiser le trafic urbain. Ils ajustent les feux en temps réel, minimisant les embouteillages et améliorant la circulation. Notre recherche se concentre sur l'optimisation des systèmes de feux de circulation pour améliorer la fluidité du trafic urbain. Après une étude de l'état de l'art sur les systèmes de feux de signalisation et les méta-heuristiques, nous proposons une approche pour ajuster les signaux en temps réel, réduisant ainsi les embouteillages et la consommation de carburant. Cette approche vise à résoudre le problème NP-difficile de la planification des feux de signalisation, avec l'objectif est de minimiser les files d'attente et les embouteillages.

Mots clés : feux de signalisation, transport, réseau routier, méta-heuristique, minimisation des files d'attente.

Remerciements

Nous tenons tout d'abord à exprimer notre gratitude infinie envers ALLAH le Tout-Puissant, qui nous a accordé la patience, la santé et la détermination nécessaires pour accomplir ce travail.

Nous souhaitons exprimer notre profonde reconnaissance à notre encadrante, Dr. KOUIDRI Siham, ainsi qu'à notre Co-encadrante, Dr. KOUIDRI Chaima, du département d'informatique, université de Saïda.
Leurs conseils avisés, leur soutien indéfectible et leur disponibilité constante ont grandement contribué à notre réussite. Leur confiance, leurs encouragements et leur assistance précieuse tout au long de notre parcours universitaire nous ont été d'une aide précieuse.

Nous tenons également à remercier chaleureusement les membres du jury du département d'informatique.
Leur engagement à évaluer notre travail avec rigueur et bienveillance est pour nous un honneur.

Enfin, nous remercions tous les enseignants et personnels administratifs de l'université de Saïda, en particulier ceux du département d'informatique, qui ont contribué à notre formation académique.
Leurs enseignements, leur soutien logistique et leur encouragement ont été essentiels tout au long de notre parcours universitaire.

Dédicaces

Je dédie ce modeste travail

À mes chers parents qui ont contribué à ma réussite et m'ont encouragés

À mon frère et mes sœurs.

À toute ma famille.

À tous ceux qui me sont chers.

Dédicaces

Je dédie ce modeste travail

À ma mère, la femme la plus chère du monde, la source de tendresse qui a tout donné sans rien recevoir, je la remercie du fond de mon coeur.

À mon père , qui a toujours cru en moi, et a mis à ma disposition tous les moyens nécessaires pour que je réussisse dans mes études.

À mes chères soeurs .

À mon cher frère.

À toute ma famille.

À toutes mes chères amies et collègues du Master 2 Réseaux

À tous qui ont contribué de prés ou de loin à la réalisation de ce modéste travail.

Table des matières

Table des figures

Liste des tableaux

Abréviations & Notations

TRANSYT	TRAfic Network Study Toole
TRRL	Transport and Road Research Laboratory
SCOOT	Split Cycle and Offset Optimisation Technique
SCATS	Sydney Co-ordinated Adaptative Trafic System
PRODYN	PROgrammation DYNamique
CERT	Centre d'Etude et de Recherche de Toulouse
ATCS	Systèmes Adaptatifs de Contrôle du Trafic
STI	Système de Transport Intelligent
TRRL	Transport and Road Research Laboratory
TRL	Traffic Research Laboratory
ATIS	Automatic Terminal Information Service
SIAD	System d'Information d'Aide au Déplacement
GA	Genetic Algorithm
SO	Snake Optimisation
PSO	Optimisation par Essaim de Particules
TLBO	Teaching-Learning-Based Optimization
ACO	Ant Colonie Optimization
SA	Simulated Annealing (Recuit Simulé)
GSA	Algorithme de Gravité de la Surface
HS	Harmonie Recherche
ACA	Algorithme des Colonies de Fourmis
XGboot	eXtreme Gradient Boosting
SDK	Software Development Kit
JVM	Java Virtual Machine
JDK	Java Development Kit
SUMO	Simulation of Urban Mobility
OPL	Open Public License
OSM	Open Street Map
XML	Extensible Markup Language
SUMO-GUI	Graphix user interface
TraCI	Traffic control interface
CSV	valeurs séparées par des virgules (comma-separated values)

Introduction Générale

Contexte

Depuis plusieurs décennies, vu l'amélioration du niveau de vie moyen, on assiste à une augmentation sans précédent des besoins en déplacement au niveau de la majorité des grandes villes à travers le monde.

La réponse à cette demande s'interprète principalement par la multiplication du nombre de véhicules en circulation. Ce phénomène a des effets sur les déplacements quotidiens des citadins et a de nombreux impacts négatifs en termes de qualité de vie. Pour assurer la fluidité du trafic et pour remédier au problème de la congestion, différentes solutions ont été proposées, comme par exemple, construire des sens giratoires, installer des barrages, réaliser des sens uniques et des autoroutes [1].

Problématique

Pour réduire la congestion dans les zones urbaines, des différentes stratégies sont adoptées telles que : la construction de nouvelles infrastructures, l'élargissement des routes très fréquentées, la mise en place des établissements publics de transport urbain et semi urbain et la mise en œuvre des stratégies de transport (par exemple d'optimiser l'utilisation des infrastructures existantes) [2].

Mais, ces solutions sont encore loin pour faciliter une gestion optimale du trafic parce qu'ils sont incapables de répondre à la fluctuation du débit de la demande. Alors la meilleure solution pour la réduction du phénomène de la congestion urbaine sera de rechercher dans la signalisation des feux.

Le contrôle des feux de signalisation est un problème complexe qui peut être formulé comme un problème d'optimisation, Cela implique de prendre en compte de nombreux facteurs, tels que le nombre de voitures à une intersection à un moment donné, la direction dans laquelle elles se déplacent, le temps qu'il faut pour qu'une voiture traverse l'intersection, et bien d'autres. De plus, ces facteurs peuvent changer dynamiquement au fil du temps, ce qui ajoute à la complexité du problème.

En raison de cette complexité, des techniques d'optimisation heuristique et méta-heuristique sont souvent utilisées pour résoudre ce problème. Ces techniques

cherchent à trouver une solution “suffisamment bonne” en un temps raisonnable, plutôt que de chercher la solution optimale parfaite qui pourrait prendre un temps prohibitif à trouver.

Objective

Dans le cadre de ce travail, notre objectif est d’appliquer une approche méta-heuristique hybride, combinant l’optimiseur Snake (SO) avec l’algorithme génétique(GA), pour réguler les feux de signalisation.

Nous nous concentrons en particulier sur l’intersection du village Boudia, qui est considérée comme l’une des plus congestionnées de la ville de Saida.

En utilisant cette méthode méta-heuristique hybride, nous visons à obtenir une programmation optimale globale des feux de signalisation. L’objectif ultime est de minimiser les files d’attente aux feux, afin d’améliorer la fluidité du trafic routier, de réduire les embouteillages, la consommation de carburant et les émissions de gaz à effet de serre, tout en améliorant la sécurité routière.

Notre travail vise donc à utiliser des techniques d’optimisation avancées pour résoudre des problèmes de trafic complexes et contribuer à des villes plus durables et plus sûres.

Organisation du manuscrit

Le travail que nous avons réalisé dans le cadre du problème est résumé dans ce document, qui est structuré en quatre chapitres encadré par une introduction et conclusion générale :

Le premier chapitre est dédié à un aperçu général sur le trafic. Il présente un rappel de quelques concepts de base sur les variables de trafic et la classification des modèles.

Le second chapitre présentera les différentes méta-heuristiques existantes, leur principe de fonctionnement, leurs avantages/inconvénients, et leur utilisation pour résoudre des problèmes d’optimisation complexes comme celui des feux de circulation.

Dans le chapitre trois, nous exposons l’approche d’optimisation appliquée dans la cadre de notre travail, à savoir son principe, ses variables, la fonction d’évaluation utilisée, ...etc.

Les expérimentations réalisées et les résultats obtenus par l’approche proposée sont exposées et analysées en dernier chapitre.

Enfin, Une synthèse et un ensemble de perspectives feront l’objet de notre conclusion.

Chapitre 1

État de l'art sur la gestion des feux tricolores

1.1 Introduction

Un réseau routier est conçu afin de permettre à ses usagers de se déplacer d'un point à un autre. Dans ce contexte, il est composé d'un ensemble de routes avec plus ou moins de voies, en fonction des zones d'activités et des lieux d'habitation. Les croisements entre les routes étant inévitables en milieu urbain, de par leur concentration, des intersections permettent de gérer les flux de véhicules entrant en conflit, pour leur sécurité et afin d'éviter les inter-blocages et de permettre aux automobilistes de changer de route. Ces intersections, ou dans certains cas des carrefours giratoires, régulent le trafic et appliquent généralement des règles de priorité à droite ou sont équipées de feux de circulation pour gérer les situations particulièrement dangereuses.

Dans ce chapitre, nous donnons en premier lieu les notions de base des feux tricolores et leurs significations. Ensuite, nous abordons l'histoire, l'évolution et les domaines d'application. Nous présentons également quelques systèmes existants pour la gestion des feux tricolores, ainsi que des outils théoriques de gestion du trafic routier et des systèmes de transport intelligents. En conclusion, nous évoquons les algorithmes de plus court chemin dans le domaine du transport et quelques inconvénients de ces systèmes.

1.2 Définition des feux de circulation tricolores

Un feu de circulation routière est un dispositif permettant la régulation du trafic routier entre les usagers de la route, les véhicules et les piétons. Les feux destinés aux véhicules à moteurs sont généralement de type tricolores, auxquels peuvent s'ajouter des flèches directionnelles. Ceux destinés aux piétons sont bicolores et se

distinguent souvent par la reproduction d'une silhouette de piéton. Les feux tricolores pour cyclistes se distinguent par la reproduction d'une bicyclette.

Généralement, un feu tricolore est composé d'un système électronique commandé. Il est composé de trois couleurs principales. La couleur rouge indique l'obligation d'arrêt aux véhicules. La couleur orange, qui ne dure que quelques secondes, signale le passage du rouge au vert. La couleur verte indique aux véhicules qu'ils ont la priorité exclusive pour passer. Ces couleurs ont été choisies parce qu'elles ont l'avantage d'être très distinctes [3].

Le système des feux de signalisation est le plus efficace pour la gestion du trafic, car il évite tout malentendu entre les différents conducteurs au moment du passage. Cependant, c'est aussi le système qui génère le plus de retard, puisqu'il favorise à chaque instant un ou deux flux (qui ne se croisent pas) et oblige systématiquement l'arrêt de tous les autres flux entrants [4].

1.3 Terminologie

Un flux de véhicules : est l'ensemble des véhicules entrant par une voie donnée et ressortant par une autre. Le trafic dans une intersection est constitué d'un ensemble de flux de véhicules, chacun provenant d'une source. Deux flux sont cohérents s'ils peuvent évacuer l'intersection simultanément. Une intersection gérée par des feux de signalisation est composée de plusieurs feux tricolores, implantés sur les différentes voies entrantes dans l'intersection [4].

Un cycle d'un feu : représente la durée qui sépare deux phases identiques de l'intersection. Il est défini par une séquence de phases.

Une phase d'un feu : est une période durant laquelle un ou plusieurs flux cohérents sont admis dans le carrefour [4].

Un carrefour : est défini comme étant une intersection de plusieurs rues où différents flux de véhicules doivent circuler de manière ordonnée. Un ensemble de carrefours constitue un réseau, chaque carrefour possède un cycle [5].

1.4 Histoire de l'évolution des feux de circulation

Au début des années 1900 [6], le monde se développait à un rythme très rapide et, avec la croissance de l'industrialisation, les villes devenaient de plus en plus surpeuplées. En outre, avec l'invention de l'automobile, le trafic sur les routes a considérablement augmenté, ce qui a rendu nécessaire un meilleur système de circulation.

En 1912, un policier américain, Lester Wire, préoccupé par l'augmentation du trafic, a eu l'idée du premier feu de signalisation électrique. Conformément à la conception de Wire, les phares ont été installés pour la première fois à Cleveland, dans l'Ohio, le 5 août 1914, à l'angle de la 105e et de Euclid Avenue.

En 1914 : le premier signal électrique à commande manuelle arrive à Cleveland. Comme le sémaphore de 1910 [7], le premier signal électrique utilisait des mots. Cependant, les mots n'étaient plus écrits sur les bras qui se soulevaient et tombaient.

Les mots «Stop » ou «Move » étaient allumés. Les poteaux s'illuminaient sur chacun des quatre poteaux d'angle autour d'une intersection. Ce n'était pas automatisé, cependant. Une cabine avec un opérateur était nécessaire pour actionner les commutateurs.

Le système permettait aux policiers de passer du centre de la rue à un coin. De ce point de vue, un officier pourrait surveiller la foule. Si un véhicule d'urgence arrivait, l'agent pourrait actionner un commutateur et dégager l'intersection en allumant tous les feux rouges. Le véhicule d'urgence pourrait maintenant passer sans effort.

Des signaux tricolores, actionnés manuellement depuis une tour située au milieu de la rue, ont été installés à New York en 1918.

En 1920 [22], un policier du nom de William Potts, à Detroit, dans le Michigan, a inventé les premiers feux de circulation à quatre couleurs et à trois couleurs. Outre le rouge et le vert, une troisième couleur orange (ou jaune) a été introduite. Detroit est devenue la première ville à avoir mis en place des feux tricolores.

Dans les années 1920, plusieurs feux de signalisation automatisés ont été installés dans les principales villes du monde. Le feu de signalisation moderne utilise toujours ce célèbre modèle en forme de **T** avec trois couleurs différentes.

Les premières lumières de ce type apparaissant en Grande-Bretagne se trouvaient à Londres. Sur la jonction entre St James Street et Piccadilly, en 1925. Ils étaient actionnés manuellement par des policiers à l'aide d'interrupteurs. Des signaux automatiques, fonctionnant sur un intervalle de temps, ont été installés à Wolverhampton, en 1926.

Les premiers signaux activés par un véhicule en Grande-Bretagne ont eu lieu à la jonction entre Gracechurch Street et Cornhill on the City, en 1932.

Les signaux normalisés rouge-orange-vert sont maintenant universellement adoptés [8].

1.5 Domaine d'emploi

L'emploi des feux de circulation a pour but d'assurer la sécurité des piétons et des usagers des véhicules, ainsi que d'améliorer la fluidité de la circulation [9].

Voici quelques exemples d'utilisation :

Gestion du trafic aux intersections : Les feux de signalisation régulent le flux de véhicules à des carrefours, permettant une circulation ordonnée.

Traversée des piétons : Les feux piétons indiquent quand il est sécurisé pour les piétons de traverser la route, exploitation par sens uniques alternés, dans des sections où le croisement est impossible ou dangereux (comme des ouvrages d'art étroits). Les feux de circulation gèrent la circulation en alternant les sens uniques.

Affectation des voies d'une chaussée : Les feux peuvent affecter certaines voies à un sens de circulation en fonction des besoins ou les condamner temporairement.

Contrôle d'accès aux voies rapides : Les feux gèrent l'accès aux autoroutes et voies rapides, gestion des points de contrôle, aux péages et les feux contrôlent l'arrêt des véhicules.

Protection contre les obstacles intermittents : Les feux assurent la sécurité lors de passages à niveau, de traversées réservées aux transports en commun, de ponts mobiles, d'atterrissages d'avions, d'avalanches, etc.

1.6 Systèmes existants de gestion des feux tricolores

Certains modèles de gestion des feux de circulation ont été commercialisés. Ceux-ci utilisent soit des capteurs qui déterminent la position des véhicules en temps réel, soit une méthode qui permet de simuler le déplacement des véhicules, ou bien les deux.

1.6.1 TRANSYT

Le premier modèle commercialisé a été TRANSYT (Trafic Network Study Tool). C'est un programme d'optimisation de la commande des feux en temps fixe [8]. Pour un réseau comprenant un certain nombre de tronçons et de carrefours, et pour une période caractéristique pendant laquelle les débits entrants dans le réseau sont considérés constants, le programme détermine les plans de feux (répartitions optimales de durées de vert entre les différentes branches de tous les carrefours et décalages optimaux entre ces mêmes carrefours) conduisant à un fonctionnement optimal du réseau. Tous les feux du réseau fonctionnent sur la même durée de cycle.

La première version de TRANSYT date de 1967 et a été développée par le TRRL (Transport and Road Research Laboratory) en Grande-Bretagne. Depuis, les plans de feux calculés par TRANSYT sont à la base de nombreux systèmes

de régulation de trafic implantés dans de nombreuses villes britanniques et à travers le monde. TRANSYT est également devenu un système de référence pour l'évaluation de l'efficacité des systèmes de régulation temps réel. Il continue d'tre amélioré, sa dernière version (version 10 new release) date de 1996. À partir de la représentation du réseau et des données d'entrée caractérisant.

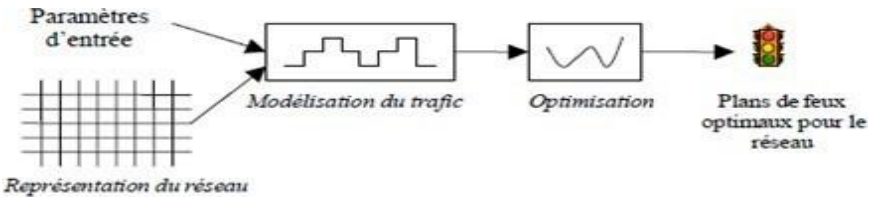

FIGURE 1.1 – Architecture de TRANSYT

L'écoulement du trafic sur le réseau est évalué par le modèle d'écoulement du trafic, qui calcule un indice de performance global du réseau. Le module d'optimisation cherche ensuite les plans de feux qui minimisent cet indice.

1.6.2 SCOOTT

SCOOT (Split Cycle Offset Optimization Technique) est un système de contrôle à la fois réactif et adaptatif, entièrement centralisé, développé par le TRL (Traffic Research Laboratory), un centre de recherche anglais sur les transports. À l'aide de détecteurs placés sur le terrain, SCOOT se base notamment sur un indice de performance afin de générer des plans de feux en fonction de la demande des utilisateurs.

Cet indice est calculé par rapport au délai d'attente moyen, à la longueur des files d'attente et aux arrêts sur le réseau.

Cet aspect dynamique est réalisé à l'aide d'un aller-retour régulier de mesures et de décisions entre les équipements sur le terrain et un centre de contrôle. Cette centralisation et ce suivi régulier de la circulation impliquent un passage à l'échelle limité, car de gros besoins en calcul sont nécessaires et car tous les détecteurs doivent être interconnectés. Cela limite leur déploiement aux plus grands carrefours.

1.6.3 SCATS

SCATS (Sydney Coordinated Adaptive Traffic System) [10] a été initialement développé pour Sydney et d'autres villes australiennes. Il est entièrement adaptatif et utilise une notion d'hiérarchie, ce qui forme une certaine distribution sur le réseau. Entre le recueil des données sur le terrain et le centre de contrôle, des contrôleurs intermédiaires sont insérés, permettant d'alléger la charge globale du système et d'avoir un contrôle découpé en plusieurs zones, l'ensemble des acteurs utilisant des communications synchronisées.

De manière similaire à SCOOT, ce système ajuste le temps des cycles et d'autres paramètres en fonction des données recueillies afin de diminuer le délai et les arrêts, mais n'utilise pas la même stratégie. Les valeurs recueillies permettent la sélection de plans de feux parmi une large librairie, sur lesquels le système va se baser pour proposer des plans adaptés. De plus, contrairement à SCOOT, les détecteurs sont uniquement placés au niveau des feux de circulation.

Ce système de régulation s'appuie sur des bibliothèques séparées de durées de cycle, de décalages et de durées de vert, ainsi que sur un algorithme temps réel de reconstitution du plan de feux. Le plan de feu est ainsi reconstitué et non stocké tel quel dans une bibliothèque. Figure 1.2

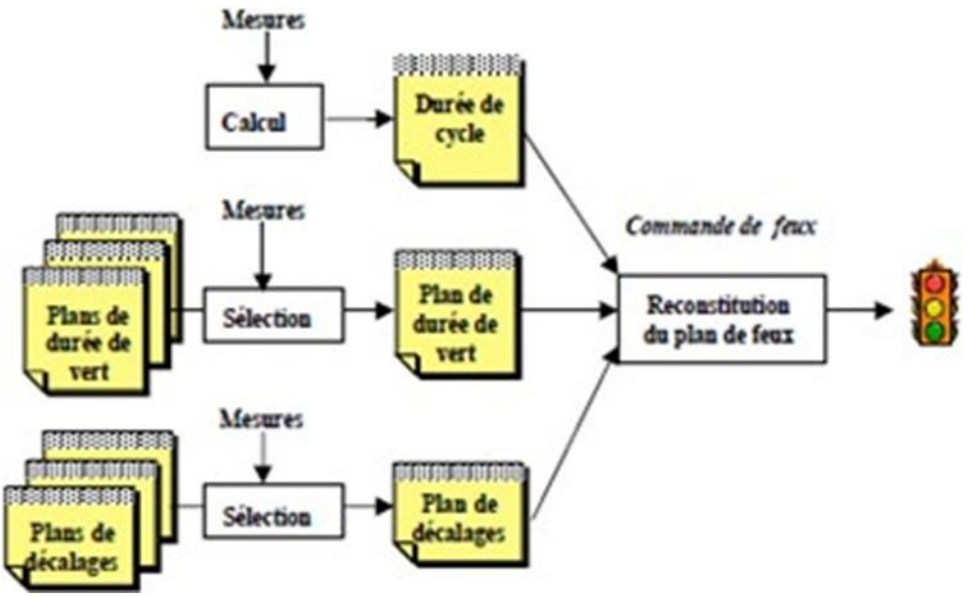

FIGURE 1.2 – Architecture de SCATS

Ce système de régulation ne comprend pas de module d'écoulement du trafic : son fonctionnement ne repose que sur la disponibilité de données explicites décrivant le trafic. L'objectif général est de minimiser les retards et les arrêts en choisissant les paramètres de base du système de régulation du trafic, tels que la durée des feux verts, les décalages et la durée du cycle. La régulation se décompose en deux niveaux : une régulation stratégique sur des ensembles de carrefours et une régulation tactique au niveau de chaque carrefour.

1.6.4 PRODYN

PRODYN (PROgrammation DYNamique) [12] est un système décentralisé et adaptatif au trafic développé par le CERT (Centre d'Étude et de Recherche de Toulouse) en France dans les années 1980. Dans ce système, 2 à 3 boucles électromagnétiques sont disposées sur chaque tronçon, et l'état supposé du trafic sur chaque

voie est estimé à l'aide d'un modèle d'écoulement simple permettant d'anticiper la progression des véhicules sur la voie.

Ce système réalise une optimisation sur l'intersection «isolée » Ṫ outefois, certaines versions de ce système permettent une communication entre les intersections voisines afin d'anticiper les flux entrants. La stratégie utilisée par ce système consiste à analyser à chaque pas de temps (de 5 secondes) si commuter l'état du feu (c'est-à-dire changer de phase) est la décision optimale, c'est-à-dire si elle minimise le temps d'attente des véhicules devant l'intersection pour les 75 prochaines secondes d'après le modèle d'écoulement utilisé.

1.6.5 Outils théoriques de la gestion de trafic routier

Il est courant pour des modèles dynamiques de la littérature de se servir d'outils théoriques, parfois en faisant un rapprochement à la réalité, parfois sans aucune notion physique (technologie utilisée, disposition). Dans les systèmes de gestion de trafic, plusieurs outils théoriques sont étudiés, répandus dans la littérature et servant de base à certains modèles étudiés.

Contrôle par la logique floue

La logique floue permet de mettre en place des degrés dans la vérification d'une condition, et non plus se borner à un choix strictement binaire. Ce principe est utilisé par quelques auteurs pour traiter le problème de la gestion des feux de circulation et permet de simplifier le problème, ce qui change des méthodes d'optimisation mathématique habituelles souvent lourdes.

Quelques exemples de travaux peuvent être trouvés au travers de [13] [14]. Nous pouvons également citer [15] qui utilise la logique floue afin de déterminer le temps d'un feu en fonction du nombre de véhicules présents sur les voies : à un nombre de véhicules correspond un intervalle définissant une durée de feu (exemple : moins de 5 véhicules par minute donne le feu vert pendant 10 secondes).

Ce principe apparaît comme idéal à utiliser :

— Théorie simple s'appliquant à des problèmes complexes.

— Robustesse de la commande floue par rapport aux incertitudes.

Les inconvénients sont tout de même importants : les techniques de mise en place et les réglages sont empiriques et aucune théorie ne permet de démontrer la stabilité et la robustesse d'une telle méthode [16].

1.7 Systèmes de transport intelligents

La recherche au niveau de la gestion des feux de circulation a connu une très grande popularité depuis les années 1960 jusqu'aux années 1980. Après cette période, ce domaine de recherche a été un peu délaissé. L'arrivée des systèmes de

transport intelligents (STI) a relancé l'étude de la gestion des feux de circulation en permettant l'utilisation de l'information sur l'évolution de l'état de la circulation aux carrefours. Ceci a amené la possibilité d'optimiser les décisions de communication à un carrefour donné en fonction de ces informations.

Dans les systèmes décentralisés, certaines informations peuvent éventuellement être échangées d'un carrefour à l'autre, mais généralement la majorité d'entre eux ne tiennent compte que de l'information provenant du carrefour local pour leur optimisation. Les STIs centralisés relient les différents carrefours du réseau entre eux afin de rendre possible l'échange d'information.

Le partage de ces informations permet alors d'optimiser les décalages des décisions de communication entre les carrefours en fonction du changement des conditions de trafic, ce qui permet une meilleure qualité de gestion de la circulation, mais au prix d'une complexité beaucoup plus élevée que pour les systèmes décentralisés. Comme on peut le voir, les possibilités qu'amène l'utilisation des STIs sont nombreuses et débouchent sur de nouveaux problèmes d'optimisation [5].

Les systèmes de feux adaptatifs et semi-adaptatifs sont tous les deux des systèmes qui utilisent des données en temps réel. Ces systèmes peuvent détecter la présence de véhicules d'un flux donné dans un carrefour et leur donner priorité au feu vert. Un système adaptatif détecte les véhicules sur tous les flux du carrefour et peut donc s'ajuster à la demande sur l'ensemble du carrefour. Le système semi-adaptatif n'a des détecteurs que sur les artères principales. Il ne s'ajuste donc qu'à la demande de ces dernières. Les artères secondaires, ayant une demande moins élevée, ne sont généralement pas équipées de détecteurs. Leur temps de vert sera donc ajusté en fonction des conditions observées dans les artères principales et non des leurs, ce qui peut réduire la fluidité de la circulation pour ces approches.

1.8 Algorithmes du plus court chemin dans le domaine du transport

Les algorithmes de plus court chemin sont énormément utilisés dans le domaine du transport, spécialement pour deux applications du transport public :

— **Analyse des performances des réseaux de transport :** Une phase de modélisation du réseau et d'analyse de performances doit être établie avant toute conception de réseau de transport. Cette analyse permet de faire un choix sur l'emplacement des lignes de transport, des arrêts et des stations.

— **Aide au déplacement et planification des itinéraires dans les systèmes d'information avancés (ATIS) :** Ceci revient à implémenter des algorithmes de plus court chemin sur les systèmes d'information d'aide au déplacement

(SIAD) qui permettent de répondre à des requêtes locales d'itinéraires provenant de différents utilisateurs.

Le problème de plus court chemin consiste à trouver le chemin qui minimise le coût global en partant d'un nœud de départ x vers un nœud d'arrivée y dans un graphe $G = (N, E)$, où $x, y \in N$ et les poids des arcs représentent le coût. Dans la littérature, cette problématique a fait l'objet de nombreuses recherches. Plusieurs algorithmes ont été mis en place pour traiter les spécificités des différents types de graphes (graphes cycliques, graphes avec valeurs négatives, etc.).

Ces algorithmes sont souvent classés en deux catégories : les algorithmes de correction d'étiquettes et les algorithmes de fixation d'étiquettes. Il existe plusieurs algorithmes pour trouver le plus court et meilleur chemin, tels que l'algorithme de Bellman-Ford-Moore, l'algorithme de Prim, l'algorithme de Kruskal et l'algorithme de Dijkstra...

1.9 Inconvénients des feux tricolores

Les feux de signalisation jouent un rôle non négligeable dans les émissions de CO_2 en ville. Les rejets de gaz carbonique triplent lorsque les véhicules sont bloqués dans des files créées par des feux de signalisation non synchronisés.

Les véhicules doivent attendre jusqu'à 3 cycles dans certaines rues, et la perte de temps peut aller jusqu'à 5 minutes.

Un autre inconvénient est que la concentration de particules fines augmenterait pour le monoxyde de carbone CO et pour l'oxyde d'azote NO_x.

1.10 Conclusion

Dans ce chapitre, nous avons survolé de manière générale les feux tricolores. Nous avons défini ces derniers, leur évolution et quelques systèmes de gestion. Ensuite, nous avons présenté les outils théoriques de la gestion du trafic routier et les systèmes de transport intelligents.

Enfin, nous avons terminé par les algorithmes de plus court chemin dans le domaine du transport et quelques inconvénients des feux tricolores.

Dans le prochain chapitre, nous allons decrire les differentes meta-heuristiques proposer pour la gestion des feux de signalisations.

Chapitre 2

État de l'art sur les meta-heuristiques

2.1 Introduction

Les méthodes d'optimisation visent à trouver des solutions optimales ou quasi optimales aux problèmes et peuvent être divisées en méthodes exactes et méthodes (méta)heuristiques. La différence entre l'heuristique et la méta heuristique réside dans le fait que la première exploite les propriétés du problème, tandis que la seconde est générale et peut être appliquée à n'importe quel problème. Ce chapitre passe en revue les principes des métas heuristiques, un outil important pour résoudre des problèmes d'optimisation complexes du monde réel.

2.2 Les meta-heuristiques

2.2.1 Définition des meta-heuristiques

Une meta-heuristique est une méthode générale qui vise à trouver l'optimum global (c'est-à-dire l'extremum global d'une fonction) d'un problème en explorant l'espace de recherche sans se laisser piéger par les optimums locaux [17].

Ce terme a été introduit par Fred Glover, le concepteur de la recherche taboue, qui est l'une des nombreux métas heuristiques existantes.

Contrairement aux méthodes traditionnelles qui peuvent être limitées par la taille ou la complexité des problèmes, les méta-heuristiques offrent une approche flexible et efficace pour trouver des solutions de qualité acceptable dans des délais raisonnables.

2.2.2 Intérêt des meta-heuristiques pour résoudre des problèmes complexes

Les méta-heuristiques représentent des approches d'optimisation qui exploitent une dose d'aléatoire et de probabilité afin de découvrir des solutions (presque) optimales pour des problèmes complexes pour lesquels aucune méthode classique plus efficace n'est connue. Ces problèmes peuvent revêtir diverses formes, tels que des problèmes combinatoires, multi-objectifs, dynamiques, ou assortis de contraintes.

L'attrait des méta-heuristiques réside dans leur caractère généralement générique, ce qui signifie qu'elles sont adaptables à une grande variété de problèmes sans nécessiter des modifications majeures dans l'algorithme sous-jacent. Souvent inspirées par des phénomènes naturels, des principes physiques ou des comportements humains, elles bénéficient d'une certaine intuition et d'une robustesse intrinsèque.

Ces méthodes sont capables d'explorer efficacement l'espace de recherche en alternant entre l'exploration, qui consiste à sonder de vastes zones de l'espace pour générer une diversité de solutions, et l'exploitation, qui consiste à examiner de manière plus approfondie les zones prometteuses pour identifier des solutions de meilleure qualité. Cette approche permet d'éviter de rester piégé dans des optima locaux, qui sont des solutions ne pouvant être améliorées localement mais ne représentant pas les solutions optimales.

En outre, les méta-heuristiques sont flexibles et modulables, ce qui leur permet d'être combinées ou hybridées entre elles, ou encore avec d'autres techniques, afin d'améliorer leur performance ou leur adaptabilité. Elles peuvent également bénéficier de mécanismes d'apprentissage pour s'ajuster aux particularités du problème ou aux évolutions de l'environnement.

Leur efficacité a été démontrée dans de nombreux domaines tels que la planification, l'ordonnancement, le routage, la conception et la bio-informatique, ce qui en fait des outils puissants et polyvalents pour résoudre une grande diversité de problèmes complexes.

2.2.3 Les avantages et les inconvénients des meta-heuristiques

Les avantages des méta-heuristiques sont les suivants [18][19] :

— Elles sont généralement génériques, c'est-à-dire qu'elles peuvent s'adapter à une large gamme de problèmes différents, sans nécessiter de changements profonds dans l'algorithme employé.

— Elles sont souvent inspirées de la nature, de la physique ou de l'humain, ce qui leur confère une certaine intuition et une certaine robustesse.

— Elles sont capables d'éviter de tomber dans des optima locaux, qui sont des solutions qui ne peuvent pas être améliorées par des modifications locales, mais qui ne sont pas les meilleures solutions possibles.

— Elles peuvent être enrichies par des mécanismes d'apprentissage, qui leur permettent de s'ajuster aux caractéristiques du problème ou aux changements de l'environnement.

Les inconvénients des méta-heuristiques sont les suivants[18][19] :

— Elles ne garantissent pas la solution optimale, mais seulement une solution de bonne qualité, qui peut dépendre de l'initialisation, des paramètres ou du hasard.

— Elles nécessitent souvent un réglage fin des paramètres, qui peut être difficile à réaliser et qui peut influencer fortement la performance de l'algorithme.

— Elles peuvent être sensibles au choix de la représentation du problème, de la fonction objective, de la fonction de voisinage, des opérateurs de recherche, etc.

— Elles peuvent être coûteuses en temps de calcul, surtout si le problème est de grande taille ou si la fonction objectif est complexe à évaluer.

2.3 Techniques d'optimisation meta-heuristiques

Les techniques d'optimisation méta heuristique sont des méthodes d'optimisation stochastique qui utilisent un certain degré d'aléatoire et de probabilité pour trouver les solutions (presque) optimales. Elles sont adaptées aux problèmes de type (je le reconnais quand je le vois).

Dans ces problèmes, nous n'avons pas d'information préalable sur l'aspect de la meilleure solution.

Quand nous disposons d'une solution candidate, sa qualité ou sa pertinence peut être évaluée en utilisant la fonction objective.

Les algorithmes méta heuristiques peuvent être classés de différentes manières; l'une des catégories les plus populaires est basée sur le nombre de solutions traitées

à chaque itération. Les algorithmes basés sur une seule solution (S-based) sont des algorithmes qui manipulent une solution à chaque itération du processus d'optimisation, tandis que les algorithmes basés sur une population (P-based) manipulent un ensemble de solutions (appelé population) à chaque itération du processus d'optimisation.

Le recuit simulé (SA), la recherche tabou (TS) et le grand déluge (GD) sont des exemples d'algorithmes méta heuristiques S-based. L'algorithme génétique (GA), la colonie d'abeilles artificielles (ABC), l'optimisation par colonies de fourmis (ACO), l'optimisation par essaims de particules (PSO) sont des exemples d'algorithmes méta heuristiques (P-based).

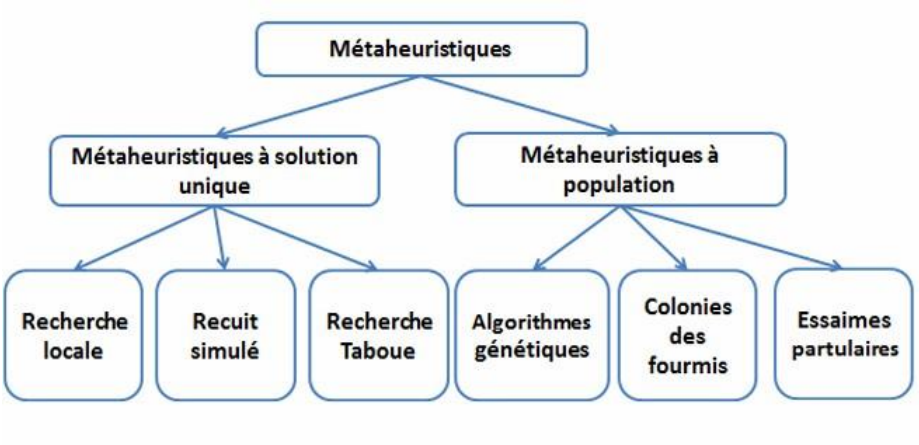

FIGURE 2.1 – Classification des meta-heuristiques [20]

De plus, selon la nature de l'inspiration, où la plupart des algorithmes méta heuristiques basés sur une population sont inspirés de la nature, ils peuvent être classés en quatre grands groupes : basés sur l'évolution (par exemple GA, ES), basés sur l'essaim (par exemple PSO, TLBO et ACO), basés sur la physique (par exemple SA, GSA), et basés sur l'humain (par exemple HS).

Dans les algorithmes meta-heuristiques P-based, le processus d'optimisation est accompli en deux phases principales.

L'exploration (ou la diversification) et l'exploitation (ou l'intensification).

Dans l'exploration, une grande échelle de régions de l'espace de recherche est examinée pour générer des solutions diverses afin de réduire la chance de tomber dans un minimum local.

D'autre part, l'exploitation consiste à examiner les régions prometteuses plus attentivement pour trouver de meilleures solutions. Cependant, un compromis approprié entre ces deux composantes est nécessaire pour atteindre l'optimalité globale.

2.4 Présentation des principales meta-heuristiques utilisées en optimisation

Les meta-heuristiques sont des techniques d'optimisation populaires et puissantes utilisées pour résoudre une grande variété de problèmes d'optimisation.

Voici une présentation des principales meta-heuristiques couramment utilisées en optimisation :

2.4.1 meta-heuristiques à solution unique

Recuit simulé

Le recuit simulé est une meta-heuristique qui simule le processus de refroidissement lent d'un métal pour minimiser son énergie. Il accepte des mouvements dégradant la solution courante selon une probabilité dépendant d'un paramètre de température, afin d'éviter de rester bloqué dans des optima locaux.

L'idée principale du recuit simulé, proposé par Metropolis en 1953, est de simuler le comportement de la matière durant le processus de recuit utilisé en métallurgie. Le but est d'atteindre un état d'équilibre thermodynamique représentant la solution optimale du problème.

L'algorithme génère successivement des configurations à partir d'une solution initiale S_o et d'une température initiale T_o. Cette température diminue progressivement jusqu'à atteindre un état d'équilibre correspondant à l'optimum global.

Une nouvelle solution plus coûteuse que la solution courante n'est pas forcément rejetée. Son acceptation est déterminée aléatoirement selon la différence de coût et la température. Ce paramètre de température permet de prendre en compte le fait qu'au début du processus, on accepte plus facilement des solutions coûteuses pour mieux explorer l'espace de recherche. Mais plus on avance dans l'optimisation, moins on accepte des solutions dégradées ou trop coûteuses.

Ainsi, l'utilisation d'une température décroissante évite de rester bloqué dans des optima locaux et augmente les chances de converger vers l'optimum global. Le recuit simulé permet donc de résoudre le problème des minimas locaux dans les cas d'optimisation non linéaire.

Algorithme :

L'algorithme de recuit simulé peut être résumé comme suit :

recuit_simule

```
Choisir une solution initiale S
Choisir une temperature initiale Ti > 0
Choisir une temperature finale Tf > 0 // Tf < Ti
Choisir un nombre d'itération NB
(a une temperature Fairnénée)
Choisir le coefficient de diminution de la temperature F ∈ [0,1]
T <- T_{i}
TQ (Tf < T) faire
    Pour (k = 1 à NB)
        S' <- voisin aléatoire de S
        Δ <- f(S') - f(S)
        Si (Δ ≤ 0) alors
            S <- S'
        Sinon
            S <- S' avec la probabilité e^-Δ/T
        Finsi
    Finpour
FinTQ
```

FIGURE 2.2 – Pseudo code de l'algorithme du recuit simulé. [21]

Explication :

Initialement on donne le système une très haute température puis on le refroidit petit à petit. Le refroidissement du système doit se faire très lentement pour avoir l'assurance d'atteindre un état d'équilibre à chaque température T.

Le voisinage $N(s)$ d'une solution s est l'ensemble des solutions qu'on peut obtenir en déplaçant légèrement les atomes du système physique depuis l'état actuel.

A chaque itération, on génère une seule solution voisine et on la compare à la solution actuelle s. Si elle est meilleure, on la conserve, sinon on a deux possibilités :

— Si T est grande, $exp(-\triangle E/T)$ est proche de 1, donc on accepte le mouvement même s'il est mauvais.

— Si T est très petite, $exp(-\triangle E/T)$ est proche de 0, donc on rejette les mouvements qui augmentent l'énergie (la différence).

Méthode :

On tire un nombre aléatoire dans l'intervalle $[0, 1[$,
si le nombre est $< exp(-\triangle E/T)$, on conserve le mouvement, sinon on l'ignore.

La meilleure solution trouvée est stockée dans la variable s^*.

Les avantages et les inconvénients de recuit simulé :

1. **Avantages :**
   - Traite les fonctions de coût avec les degrés tout à fait arbitraires de non linéarité, la discontinuité, et l'imprévisibilité.
   - Processus assez arbitraire sur les conditions limites et les contraintes imposées sur les fonctions de coût.
   - Simple à implémenter.
   - Garantie Statistique de trouver une solution optimale.
2. **Inconvénients :**
   - Le non déterminisme.
   - Difficulté de choisir les paramètres efficaces ; par exemple le schéma de refroidissement.
   - Le compromis entre la vitesse et l'optimisation.

Recherche tabou

La recherche tabou est une méta heuristique basée sur l'utilisation d'une liste taboue pour interdire temporairement des solutions récemment explorées, éviter les cycles et favoriser l'exploration de nouvelles régions de l'espace de recherche.

La recherche tabou est une méthode d'optimisation qui repose sur l'idée de modifier une solution de départ en choisissant une solution voisine qui réduit la fonction à optimiser. Il faut noter que cette modification peut parfois détériorer la qualité de la solution. Cela permet de sortir des situations où aucune amélioration n'est possible localement.

Le problème est qu'on peut revenir sur la même situation qu'on a quittée. C'est pourquoi il faut que la méthode se souvienne des solutions visitées, le principe est de rendre interdits (d'où le nom de tabou) certains changements ou certains éléments de ces changements (par exemple, interdire les changements inverses).

Les solutions visitées sont stockées dans ce qu'on appelle la Liste Tabou dont la taille est un paramètre à régler.

Algorithme :

L'algorithme de recherche tabou peut être résumé comme suit :

Tabou

```
x := solution aléatoire
fmin := f(x)
xmin := x

TABOU := liste de solutions s(x), de longueur L
TABOU = VIDE

REPETER
    générer un N-échantillon TEL QUE s(x) ∈ voisinage S(x) et
        {x, s(x)} ∉ TABOU

    f(s(x)) = min[f(s(xi))]
        1≤i≤N

    ajouter ({x, s(x)}, TABOU)
    x := s(x)

    SI f(x) < fmin
        fmin := f(x)
        xmin := x
    FIN DE SI
JUSQU'A conditions d'arrêt satisfaites
```

FIGURE 2.3 – Pseudo code de l'algorithme de recherche tabou. [21]

Les avantages et les inconvénients de recherche tabou :

1. **Avantages :**
   Les avantages de la technique de recherche tabou se résument en deux points essentiels :
   - Elle est très performante.
   - Fonctionnement simple à comprendre.
2. **Inconvénients :**
   La méthode RT présente aussi des inconvénients, tels que :
   - Elle nécessite des paramètres difficiles à choisir.
   - Elle consomme beaucoup de ressources si la liste des tabous est trop grande.
   - Elle ne garantit pas la convergence vers une solution optimale.

2.4.2 meta-heuristiques à population de solutions

Algorithmes évolutionnaires

Les algorithmes évolutionnaires sont des méta heuristiques d'optimisation s'inspirant de mécanismes génétiques et évolutionnaires. Ils ont été initiés dans les années 1960 par John Holland, puis popularisés notamment par les travaux de Goldberg dans les années 1980.

Leur principe repose sur quatre éléments :

1. L'évaluation de l'adaptation de chaque individu via une fonction objective.
2. La sélection des meilleurs individus.
3. Le croisement par mélange de matériel génétique entre individus.
4. La mutation par modification aléatoire d'individus.

Ces algorithmes simulent l'évolution d'une population de solutions potentielles (les individus). A chaque génération, des opérateurs de sélection, croisement et mutation sont appliqués pour produire de nouveaux individus. La sélection favorise les plus adaptés, le croisement et la mutation explorent l'espace de recherche.

La population évolue ainsi vers des individus mieux adaptés au problème, se rapprochant progressivement de l'optimum global. Les critères d'arrêt sont généralement un nombre de générations, une convergence ou une stagnation.

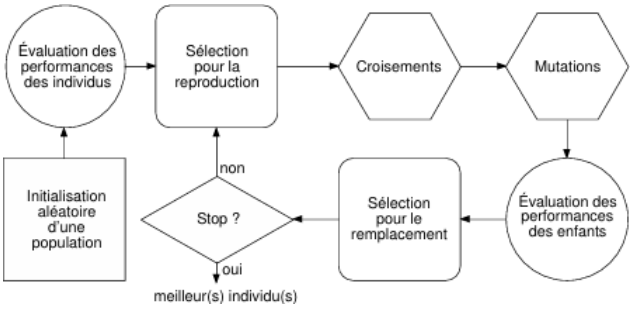

FIGURE 2.4 – Principe d'un algorithme évolutionnaire standard [22]

Ces algorithmes se distinguent des méthodes classiques par l'utilisation d'un codage des solutions, une optimisation basée sur une population, aucune régularité imposée à la fonction objective, et des règles de transition probabilistes.

Les avantages et inconvénients des algorithmes évolutionnaires :

Avantages :

— Flexibles et adaptables à divers problèmes d'optimisation.

— Capables de résoudre des problèmes complexes, non linéaires et non convexes.

— Ne nécessitent pas d'informations sur le gradient de la fonction objectif.

— Peuvent explorer efficacement de vastes espaces de recherche.

— Maintiennent une population diversifiée de solutions

Inconvénients :

— Temps de calcul souvent élevé, en particulier pour les problèmes de grande dimension.

— Réglage des paramètres (taille de la population, taux de mutation, etc.) pouvant être délicat.

— Convergence vers l'optimum global non garantie

— Performances dépendantes de la représentation des solutions et des opérateurs génétiques choisis

— Difficultés potentielles à maintenir la diversité de la population au fil des générations

Colonies de fourmis :

Les colonies de fourmis sont des métas heuristiques s'inspirant du comportement des fourmis pour la construction progressive de bonnes solutions via une communication indirecte basée sur des traces de phéromones numériques.

L'idée originale de cette méta heuristique s'inspire du comportement des fourmis réelles lorsqu'elles cherchent de la nourriture. Bien qu'individuellement peu intelligentes, elles parviennent collectivement à trouver le chemin le plus court entre leur nid et une source de nourriture.

Leur secret réside dans la stigmergie, c'est-à-dire une communication indirecte via l'environnement. Les fourmis déposent des phéromones sur leur passage et se laissent guider par ces pistes odorantes. Plus un chemin est emprunté, plus il sera marqué de phéromones et donc attractif. Mais les phéromones s'évaporant avec le temps, seuls les plus courts chemins garderont une forte intensité.

L'algorithme de colonie de fourmis artificielles, développé par Marco Dorigo, reprend ce principe. Chaque fourmi est un agent qui explore le graphe de manière probabiliste, avec une préférence pour les arêtes les plus marquées en phéromones. Après un parcours, elle renforce les phéromones sur le chemin emprunté. L'évaporation régulière évite de rester bloqué dans des optima locaux.

Avec les itérations, les phéromones s'intensifient sur le plus court chemin et disparaissent ailleurs. Le processus converge vers une solution proche de l'optimum global. Cette méta heuristique s'est étendue à d'autres problèmes d'optimisation, une fourmi représentant alors une solution potentielle se déplaçant dans l'espace de recherche.

Algorithme :
L'algorithme de colonies de fourmis peut être résumé comme suit :

ACO

```
Tantque (la condition d'arrêt n'est pas atteinte) Tantque (L'état final n'est pas
atteint)
Pour (chaque fourmi)
Choisir l'état suivant en fonction de la mémoire et de
l'environnement local
Mettre à jour les phéromones sur l'arc choisi.
FinPour
Fin TantQue
Pour (chaque fourmi)
Evaluer la solution obtenue
FinPour
Mise à jour globale des phéromones
Fin Tantque
Afficher la meilleure solution trouvée
```

FIGURE 2.5 – Pseudo code de l'algorithme colonies de fourmis [22]

Les avantages et les inconvénients des algorithmes colonies des fourmis :

Avantages :

— Rapidité de la méthode.

— Nouvelle méthode à trouver des solutions acceptables tout en évitant des convergences prématurées.

— Robuste et basée sur une population d'individus.

Inconvénients :

— Complexe à mettre en place et son paramétrage est subtile.

— Coût relativement élevé de la génération des solutions.

Essaims particulaires

L'optimisation par essaims particulaires simule le mouvement d'un essaim de particules dans l'espace de recherche pour converger vers des optima globaux, en communiquant les meilleures positions rencontrées.

Cette méta heuristique s'inspire du déplacement collectif d'un essaim d'oiseaux ou d'un banc de poissons. Elle simule le comportement d'un essaim de particules qui évoluent de façon collaborative dans l'espace de recherche pour trouver l'optimum global.

Chaque particule représente une solution potentielle et se déplace selon une vitesse dans l'espace de recherche. A chaque itération, chaque particule :

— Se déplace vers sa meilleure position déjà atteinte (composante cognitive).
— Se déplace vers la meilleure position globale atteinte par l'essaim (composante sociale).
— Ajoute une composante aléatoire pour explorer de nouvelles zones.

Les particules communiquent donc les meilleures positions qu'elles trouvent afin de guider l'essaim vers des zones prometteuses. La vitesse est ajustée à chaque itération pour équilibrer l'exploration et l'exploitation.

L'essaim converge ainsi progressivement vers des optimaux globaux. La communication de l'intelligence collective permet d'éviter de rester piégé dans des optima locaux.

Algorithme :

L'algorithme d'essaim particulaire peut être résumé comme suit :

PSO

```
// Initialiser un essaim de particules avec des positions et des vitesses aléatoires
Pour chaque particule i de 1 à N
    x_i := position aléatoire dans l'espace de recherche
    v_i := vitesse aléatoire dans l'espace de recherche
    p_i := x_i // meilleure position personnelle
    Si f(p_i) < f(p_g) // f est la fonction objectif à minimiser
        p_g := p_i // meilleure position globale
    Fin si
Fin pour

// Répéter jusqu'à convergence ou condition d'arrêt
Tant que non terminé
    Pour chaque particule i de 1 à N
        // Mettre à jour la vitesse selon la formule de PSO
        v_i := w * v_i + c_1 * r_1 * (p_i - x_i) + c_2 * r_2 * (p_g - x_i)
        // w est le coefficient d'inertie, c_1 et c_2 sont les coefficients cognitifs
sociaux, r_1 et r_2 sont des nombres aléatoires entre 0 et 1

        // Mettre à jour la position selon la vitesse
        x_i := x_i + v_i

        // Vérifier si la nouvelle position est meilleure que la précédente
        Si f(x_i) < f(p_i)
            p_i := x_i
            Si f(p_i) < f(p_g)
                p_g := p_i
            Fin si
        Fin si
    Fin pour
Fin tant que

// Retourner la meilleure position globale trouvée
Retourner p_g
```

FIGURE 2.6 – Pseudo code de l'algorithme essaim particulaire [24]

Explication :

On peut définir le voisinage d'une particule soit par la distance euclidienne entre ses coordonnées et celles des autres particules, soit par sa position relative dans l'essaim.

Chaque particule i de l'essaim a une position X_i et une vélocité V_i qui indique le sens et la magnitude de son déplacement.

Chaque particule garde en mémoire sa meilleure solution trouvée jusqu'à présent, notée P_i (personal best), ainsi que la meilleure solution de son voisinage, notée P_g (global best).

À chaque étape, chaque particule ajuste sa trajectoire en fonction d'un vecteur qui est la combinaison linéaire de sa vélocité actuelle V_i, de sa P_i et de sa P_g.

Sa nouvelle vélocité $V_i(t+1)$ est calculée selon l'équation suivante :

$$V_i(t+1) = \omega V_i(t) + C_1 r_1 (P_i(t) - X_i(t)) + C_2 r_2 (P_g(t) - X_i(t)) \qquad (2.1)$$

où $i = 1, 2, ..., N$, et N est le nombre de particules (taille de l'essaim) ; le coefficient d'inertie ω [Shi and Eberhart, 1998] permet de réguler l'effet de la vélocité précédente.

Un facteur d'inertie élevé favorise une exploration large alors qu'un facteur d'inertie faible limite la recherche à un espace restreint ; r_1 et r_2 sont deux nombres aléatoires tirés uniformément dans $[0, 1]$, C_1 et C_2 sont deux constantes qui représentent une accélération positive.

Elles correspondent à la composante cognitive (resp. sociale) du mouvement. La position de chaque particule est aussi mise à jour à chaque étape comme suit :

$$X_i(t+1) = X_i(t) + V_i(t+1) \qquad (2.2)$$

Pour éviter que les particules ne se déplacent trop vite dans l'espace de recherche, en risquant de manquer l'optimum, il peut être utile de limiter la vélocité maximale (notée Vmax), de sorte que chaque composante de V_i soit comprise dans l'intervalle $[-V_{max}, +V_{max}]$ [Eberhart et al., 1996].

Le choix de V_{max} est une opération délicate, car ce paramètre a un impact sur l'équilibre entre exploration et exploitation. L'utilisation d'un coefficient de constriction X permet de se passer de la définition de la vélocité maximale Vmax [Clerc and Kennedy, 2002].

Les avantages et les inconvénients des algorithmes d'optimisation par essaims particulaires (PSO) :

Avantages :

— Simple à implémenter et à comprendre.

— Nécessite peu de paramètres à ajuster.

— Converge rapidement vers des solutions de bonne qualité.

— Efficace pour les problèmes continus et à variables réelles.

— Peut s'échapper des optimums locaux grâce à l'interaction entre les particules

Inconvénients :

— Performances dépendantes du réglage des paramètres.

— Tendance à converger prématurément vers des optimums locaux.

— Manque de diversité dans la population à mesure que l'optimisation progresse.

— Difficulté à maintenir la diversité de l'essaim dans les problèmes de grande dimension.

— Nécessite une adaptation pour traiter les problèmes à variables discrètes.

2.5 Applications des méta-heuristiques à l'optimisation du timing des feux de signalisation dans la littérature

Dans la littérature, les méta-heuristiques ont été largement utilisées pour résoudre des problèmes d'optimisation du timing des feux de signalisation. Cette application est cruciale pour améliorer la fluidité du trafic, réduire les temps d'attente et les émissions de gaz à effet de serre, et optimiser l'efficacité des réseaux de transport urbain. Voici quelques exemples d'applications des méta-heuristiques à l'optimisation du timing des feux de signalisation :

2.5.1 Algorithmes génétiques (GA)

Leena Singh, Sudhanshu Tripathi et Himakshi Arora (2019) [25] expliquent comment les algorithmes génétiques permettent une prise de décision en temps réel et un contrôle adaptatif des signaux de circulation, en considérant plusieurs paramètres d'optimisation tels que le nombre total de véhicules sur une route et l'importance de chaque route dans l'intersection.

Les avantages de cette approche incluent une allocation efficace des ressources, une minimisation de la fonction de fitness et une amélioration globale de la fluidité du trafic. Les auteurs ont développé un émulateur de trafic pour représenter les conditions de trafic dynamiques et ont utilisé des interfaces d'application d'algorithmes génétiques MATLAB pour implémenter l'algorithme.

Les résultats expérimentaux ont montré que la stratégie de contrôle de signalisation routière en temps réel basée sur les algorithmes génétiques améliore significativement les performances des intersections par rapport aux systèmes de contrôle de temps fixe.

Paramètres :

Fonction objectif (fitness function) à minimiser

$$f = PI_1 + PI_2 + PI_3 + PI_4 \tag{2.3}$$

Où PI_i = Performance Index de la voie $i = W_i \frac{Si}{GT_i}$

Avec W_i = poids de la voie i,
S_i = nombre de véhicules sur la voie i,
GT_i = temps de vert total pour la voie i.

Contraintes :

— Longueur de cycle fixe = $70s$.

— Extensions de temps de vert (g) bornées entre 0 et $5s$.

— $g1 + g2 + g3 + g4 = 10$ (somme des extensions = $10s$).

— Temps de vert minimum fixe $G_{min} = 15s$ pour chaque voie.

Opérateurs :

— Génération initiale aléatoire de la population (extensions de vert $g1, g2, g3, g4$).

— Évaluation de la fonction objective pour chaque individu.

— Sélection des meilleurs individus (parents).

— Croisement des parents pour générer de nouveaux individus (enfants).

— Mutation aléatoire des enfants.

En somme, cet article offre une perspective intéressante sur l'utilisation des algorithmes génétiques pour optimiser la gestion du trafic dans les zones urbaines.

Huizhen Zhang, Yueer Gao, et al. L'article (2021) [26] proposent une méthode d'optimisation des feux de signalisation basée sur une fonction de Webster modifiée pour réduire les retards aux intersections.

Cette méthode utilise un algorithme génétique pour trouver le schéma de synchronisation optimal des feux, ce qui entraîne une réduction de 15,64% des retards aux intersections.

L'algorithme génétique modifié comprend des étapes telles que l'initialisation, le codage des chromosomes, la sélection, le croisement, la mutation, la pénalisation du fitness et la régénération individuelle.

Les paramètres de l'algorithme ont été optimisés pour obtenir la meilleure vitesse de convergence. Les paramètres utilisés dans l'algorithme sont les suivants :

60 individus, 80 itérations, un taux de crossover de 0,9 et un taux de mutation de 0,1.

Les opérateurs utilisés comprennent l'initialisation des individus, le codage des chromosomes, la sélection, le croisement, la mutation, la pénalisation du fitness et la régénération individuelle.

Les résultats expérimentaux montrent des résultats prometteurs, et l'étude a été soutenue par diverses subventions sans conflits d'intérêts déclarés par les auteurs.

La disponibilité des données est limitée en raison des réglementations de la circulation.

2.5.2 Algorithme colonies de fourmis (ACO)

Jiajia et Zai'en (2012) [27] ont utilisé l'algorithme des colonies de fourmis (ACO) pour optimiser une fonction objective liée au temps de cycle et à la saturation d'un carrefour d'une intersection. Ils ont utilisé le délai, le nombre de pauses et la capacité du trafic comme indice de performance.

L'algorithme Ant Colony Optimization (ACO) fonctionne en utilisant des agents simples, appelés fourmis, pour construire itérativement des solutions à un problème d'optimisation. Guidées par les phéromones, les fourmis commencent avec une solution nulle et ajoutent progressivement des composants de solution pour construire une solution complète.

Après la construction d'une solution, chaque fourmi dépose des phéromones pour chaque composant de solution, ce qui guide les autres fourmis dans leur recherche.

Les paramètres utilisés dans l'algorithme Ant Colony Optimization (ACO) pour l'optimisation de la synchronisation des feux de signalisation comprennent le nombre de fourmis (20), le nombre d'itérations (20), la vitesse de déplacement des fourmis (0.3), le coefficient q (0.85), et le coefficient Q (0.8).

Les opérateurs utilisés dans l'algorithme Ant Colony Optimization (ACO) comprennent l'initialisation des valeurs des paramètres, le calcul du fitness et des phéromones, ainsi que l'ajustement de la probabilité de transition d'état pour chaque fourmi.

Les performances de l'algorithme ACO ont été comparées à celles de l'algorithme de Webster et de GA. L'ACO s'est avéré être efficace et faisable pour résoudre le problème d'optimisation de la synchronisation des signaux.

2.5.3 Algorithme XGBoot

Olivier Audet (Ville de Montréal) et Patrice O'Carroll (Mnubo) (2019) [28] présente une approche pour prédire la circulation routière à Montréal en utilisant l'apprentissage machine. L'objectif est d'améliorer les systèmes de gestion des feux de circulation en leur fournissant des prédictions du volume de trafic et du taux d'occupation des voies. Différentes méthodes de prédiction sont comparées.

Les auteurs proposent d'utiliser l'algorithme XGBoost (eXtreme Gradient Boosting) entraîné sur des données historiques et en temps réel de détecteurs.

Les résultats montrent que le modèle est meilleur qu'une approche naïve, surtout pendant les heures de pointe. Le modèle pourra être facilement déployé sur d'autres intersections. Les prochaines étapes sont de connecter le modèle au système de gestion des feux et d'explorer l'ajout d'autres sources de données pour améliorer les prédictions.

- Différentes méthodes de prédiction de trafic sont comparées : modèles statistiques, réseaux de neurones, etc.
- Le modèle prédit 15 minutes à l'avance le volume de véhicules et le taux d'occupation des voies.
- Les prédictions sont comparées à une approche naïve qui répète les dernières valeurs.
- Le modèle d'apprentissage machine est plus performant, surtout pendant les heures de pointe en semaine.
- L'erreur moyenne absolue est jugée satisfaisante par l'équipe d'ingénierie.
- Le modèle est flexible et indépendant du tronçon, il pourra être facilement déployé sur d'autres intersections.
- Les prochaines étapes sont de connecter le modèle au système de gestion des feux et d'ajouter d'autres sources de données.

2.5.4 Algorithme SO

Cheng, Qiao, Li et Huang (2023) [29] présentent un modèle d'optimisation des feux de signalisation qui intègre la détection du trafic routier par vidéosurveillance, la prédiction des flux de trafic et l'optimisation des temps de feux grâce à un algorithme d'optimisation appelé "snake optimization" (SO).

Le modèle YOLO-X est utilisé pour détecter et compter les véhicules à partir des données vidéo. Ensuite, un modèle LSTM prédit les flux de trafic pour la prochaine période en se basant sur les données détectées et les données historiques. Enfin, l'optimisation des temps de feux est réalisée par simulation dans le logiciel VISSIM, en utilisant l'algorithme SO.

Les tests montrent que le modèle réduit le temps de retard moyen de 23,34 % par rapport aux schémas de synchronisation fixes, ce qui démontre une meilleure optimisation.

Les auteurs fournissent les paramètres et les opérateurs utilisés dans l'algorithme d'optimisation Snake Optimization (SO) :

Paramètres :

1. num (nombre d'itérations maximales)
2. pop (nombre de serpents dans la itialitiale)
3. C_f (seuil de qualité pour la chasse)
4. C_t (seuil de température pour l'accouplement)
5. C_1 (constante contrôlant la qualité de la chasse)
6. C_2 (constante contrôlant la mise à jour de position)
7. C_3 (constante contrôlant les combats et accouplements)

Les valeurs attribuées à ces paramètres pour leur étude de cas sont :

$$num = 100, pop = 100,$$
$$C_f = 0.25, C_t = 0.6, C_1 = 0.5, C_2 = 0.5, C_3 = 2$$

Opérateurs :

1. Mouvement des serpents :
   - Exploration (chasse) : Le serpent se déplace vers une meilleure position.
   - Exploitation (accouplement) : Deux serpents s'accouplent pour générer un nouvel enfant.
   - Déplacement aléatoire
2. Mise à jour de position : Après chaque mouvement, la nouvelle position est évaluée. Si elle est meilleure, le serpent s'y déplace, sinon il peut garder sa position ou se déplacer vers le meilleur serpent.
3. Reproduction :
   - Sélection : Les meilleurs serpents sont sélectionnés pour la reproduction.
   - Croisement : De nouveaux serpents (solutions enfants) sont générés par croisement des parents.
   - Mutation : Les enfants peuvent subir des mutations aléatoires.
4. Remplacement : Les serpents enfants remplacent les moins bons serpents de la population si leur qualité est meilleure.

2.5.5 Algorithme évolutionnaire de type Hillclimbing

Andrea Vogel, Christian Goerick et Werner von Seelen (2000) [30] discutent dans l'article l'utilisation d'algorithmes évolutionnaires pour optimiser les systèmes de signalisation du trafic aux intersections.

Les auteurs présentent des résultats d'expériences sur l'adaptation des paramètres et l'évolution de la structure, montrant des résultats prometteurs dans la recherche de solutions optimales.

L'importance d'un encodage soigneux du problème et d'opérateurs d'optimisation est soulignée pour obtenir des résultats d'optimisation réussis. L'objectif est d'améliorer le flux de trafic et de réduire les temps de déplacement en automatisant le processus de conception et d'optimisation des systèmes de signalisation.

L'article met en avant la structure, les paramètres et les conditions limites impliqués dans l'optimisation des signaux, ainsi que l'utilisation d'algorithmes évolutionnaires pour le processus d'optimisation.

L'introduction du conteneur d'intersection est mentionnée comme une méthode pour séparer les processus d'optimisation et d'évaluation. L'importance de l'adaptabilité et de la flexibilité dans la conception des opérations de signalisation pour répondre aux conditions de trafic changeantes est également soulignée.

Les paramètres utilisés dans l'algorithme comprennent les plans de synchronisation des feux de signalisation adaptés à la situation du trafic. Le nombre de paramètres à optimiser dépend de la structure choisie, avec au minimum un temps vert par phase à déterminer. D'autres paramètres peuvent être nécessaires pour la signalisation réactive au trafic, tels que l'adaptation du temps vert en fonction des écarts de temps entre les véhicules.

Les opérateurs de l'algorithme génèrent de nouvelles solutions à partir des meilleures solutions actuelles. Une méthode établie pour la recherche de paramètres consiste à ajouter un nombre aléatoire distribué normalement à chaque paramètre.

Le logiciel SHARK contient plusieurs opérateurs de mutation pour les problèmes d'optimisation discrets et continus, et des opérateurs spéciaux peuvent être nécessaires en fonction de la tâche et de sa surface d'erreur.

2.5.6 Multiples algorithmes

Sahar Araghi, Abbas Khosravi, Douglas Creighton et Saeid Nahavandi (2016) [31] comparent 3 algorithmes d'optimisation pour paramétrer un contrôleur de feux IT2ANFIS (un système d'inférence floue neuro-adaptatif de type 2) afin d'optimiser les temps de signalisation dans les réseaux routiers complexes.

L'étude évalue l'efficacité de trois algorithmes d'optimisation méta heuristiques (Simulated Annealing, Genetic Algorithm, Cuckoo Search) pour ajuster les paramètres du contrôleur IT2ANFIS.

Les résultats indiquent que le contrôleur IT2ANFIS optimisé par l'algorithme Cuckoo Search présente une amélioration significative des performances par rapport aux contrôleurs à temps fixe, avec une réduction d'environ 31 % du temps de déplacement des véhicules.

L'article met en lumière l'importance croissante des techniques d'intelligence artificielle dans l'optimisation des temps de signalisation pour la gestion du trafic routier.

D'après le document, les paramètres et opérateurs suivants ont été utilisés pour les différentes méthodes d'optimisation :

Cuckoo Search (CS) :

- Nombre maximum de générations : 100
- Taille de la population : 20
- Nombre de paramètres : 16
- Taille de pas (step size) : 0.1-0.9
- Probabilité de découverte d'œuf (Pa) : 0.25

Algorithme Génétique (GA) :

- Nombre maximum de générations : 100
- Taille de la population : 20
- Nombre de paramètres : 16
- Taux de mutation : 0.01
- Taux de croisement : 0.8

Recuit Simulé (Simulated Annealing - SA) :

- Nombre maximum de générations : 100
- Taille de la population : 20
- Nombre de paramètres : 16
- Nombre maximum de rejets : 20
- Nombre maximum de succès : 20
- Nombre maximum d'essais : 20
- Température initiale : 5
- Température d'arrêt : $1^{e^{-10}}$

Il a été constaté que la recherche par cucoo surpasse nettement les 2 autres algorithmes.

Syed Shah Sultan Mohiuddin Qadri, Mahmut Ali Gökçe et Erdinç Öner (2020) [32] traitent de l'optimisation de la synchronisation des feux de signalisation dans les zones urbaines, en utilisant des méthodes basées sur l'intelligence computationnelle et des algorithmes hybrides, qui combinent plusieurs techniques d'optimisation. Il souligne l'importance de prendre en compte le comportement des piétons et des conducteurs dans les modèles de trafic, ainsi que l'intérêt de considérer les scénarios avec des véhicules autonomes.

Il présente une revue de la littérature sur le sujet, en distinguant les approches multi-objectifs, les modèles basés sur l'intelligence computationnelle, les algorithmes d'optimisation tels que les algorithmes génétiques et l'optimisation par essaim, ainsi que les outils de micro simulation pour l'évaluation des solutions proposées.

Il décrit également comment utiliser les données de trafic en temps réel pour prédire la durée du cycle des feux de signalisation, ainsi que comment utiliser les outils de micro simulation et les approches basées sur l'intelligence artificielle et les méta heuristiques pour optimiser la synchronisation des feux de signalisation afin d'améliorer l'écoulement du trafic et atténuer la congestion.

Les paramètres utilisés dans les algorithmes basés sur l'intelligence computationnelle pour optimiser la synchronisation des feux de signalisation comprennent

des éléments tels que les cycles de feux, les temps de vert, les temps de rouge, les délais moyens, les capacités du réseau, les intervalles de changement, la séquence des phases, etc.

Ces paramètres sont essentiels pour ajuster et optimiser les plans de synchronisation des feux de signalisation afin d'améliorer l'efficacité du trafic et de réduire la congestion.

Ces idées mettent en lumière l'importance croissante des approches basées sur l'intelligence computationnelle pour l'optimisation de la synchronisation des feux de signalisation dans les environnements urbains, ainsi que la nécessité de prendre en compte les comportements des usagers de la route et l'utilisation de données en temps réel pour améliorer la gestion du trafic.

2.5.7 Autres méthodes

Chong Han et Qinyu Zhang (2008) [33] proposent une approche de détection et de comptage en temps réel des véhicules à une intersection à l'aide de caméras vidéo.

L'approche comprend la différenciation des images d'arrière-plan, la détection des bords, les opérations d'érosion et de dilatation pour supprimer le bruit, ainsi que des histogrammes de projection verticale pour calculer le nombre de véhicules dans chaque voie. De plus, une stratégie de contrôle adaptatif des feux de signalisation est proposée pour améliorer l'efficacité de la circulation. Les simulations montrent que le schéma de contrôle adaptatif est plus efficace que les schémas traditionnels.

L'approche pourrait contribuer à atténuer la congestion du trafic à l'avenir. Les simulations ont démontré que le système de contrôle adaptatif des feux de circulation était plus efficace pour gérer le flux de circulation et réduire les temps d'attente par rapport au système fixe traditionnel, ce qui suggère qu'il pourrait permettre de résoudre les problèmes de congestion du trafic et d'améliorer la satisfaction des conducteurs dans des applications réelles.

L'algorithme utilise plusieurs paramètres et opérateurs pour la détection de véhicules et le contrôle adaptatif des feux de signalisation. Certains des paramètres et opérateurs utilisés incluent :

1. Histogramme de projection vertical : Utilisé pour calculer la longueur des véhicules attendant dans les voies en analysant le nombre de pixels blancs de chaque composant par rapport à l'indice de colonne ().
2. Image binaire : Obtenu à partir de l'image de variation pour identifier les objets en mouvement dans la scène ().

3. Détection de bord : Utilisée pour séparer les objets des arrière-plans dans l'image de variation ().
4. Érosion et dilatation : Opérations effectuées pour supprimer le bruit et améliorer la qualité de l'image binaire ().
5. Remplissage : Utilisé pour compléter les zones d'objets identifiés dans l'image binaire ().
6. Rotation : Employée pour aligner correctement les objets identifiés dans l'image binaire ().
7. Seuil : Utilisé pour définir une plage raisonnable pour éviter les bruits et calculer la longueur de la file d'attente dans chaque voie ().

Ces paramètres et opérateurs sont essentiels pour la détection précise des véhicules et la mise en œuvre d'un contrôle adaptatif efficace des feux de signalisation dans l'algorithme proposé.

Luis Ramirez-Polo, Miguel A. Jimenez-Barros, Vladimir Varela Narváez et Carlos Parodi Daza (2022) [34] présentent une méthodologie pour optimiser le temps des feux de circulation afin de réduire le temps d'attente des véhicules dans une zone de trafic intense à Barranquilla, en Colombie.

Les auteurs ont utilisé des techniques de simulation et d'optimisation pour trouver les temps de cycle idéaux et les durées optimales de chaque couleur des feux.

Ils ont d'abord collecté des données sur le trafic et créé un modèle de simulation stochastique discret. Ils ont testé différents scénarios en faisant varier la durée de la phase verte.

Puis, ils ont développé un modèle d'optimisation dans Flexsim pour minimiser le temps d'attente moyen des véhicules, en considérant des temps de cycle différents pour chaque intersection. Les résultats montrent une réduction de plus de 30 % du temps d'attente par rapport à la situation actuelle. Cette approche combinant simulation et optimisation peut être reproduite pour d'autres systèmes de feux de circulation.

Paramètres :

— Durées des phases verte, jaune et rouge des feux à chaque intersection (variables X_1, X_3, X_5, X_7, X_9, X_{11}, X_{13} etc).

— Temps de cycle total fixe pour chaque intersection (variables C_1 et C_2).

Contraintes :

— Durées minimales (ex : 5 secondes) et maximales (ex : 120 secondes) pour chaque phase.

— Somme des durées des phases = temps de cycle total.

— Ordonnancement logique des phases (ex : vert ⩾ jaune, rouge ⩾ jaune etc.).

Fonction objectif :

— Minimiser le temps d'attente moyen des véhicules dans le système.

Titre	Auteurs	Année	Méthode d'optimisation	Avantages
Real-Time Traffic Signal Control Strategy Using Genetic Algorithm	Singh, L., Tripathi, S., Arora, H	2009	GA	Adaptative efficace des feux de signalisation mais nécessite des ressources de calcul et des données de qualité pour être pleinement efficace.
Traffic Light Optimization Based on Modified Webster Function	Zhang, H., Yuan, H., Chen, Y., Yu, W., Wang, C., Wang, J., Gao, Y	2021	GA	L'algorithme réduit les retards, améliore le service, et utilise un algorithme génétique modifié. Il a des limites liées aux données et à la fonction d'optimisation.
Ant colony algorithm for traffic signal timing optimization	Smith, J., Johnson, A	2020	ACO	Ac trouve des bonnes solutions en utilisant les phéromones. Il est robuste et évite les optima locaux, mais il peut être lent et stagné.
Traffic Signal Timing Optimization Model Based on Video Surveillance Data and Snake Optimization Algorithm	Cheng, R., Qiao, Z., Li, J., Huang, J.	2023	SO	SO est rapide, simple et efficace, qui a des applications variées. Il requiert cependant une bonne connaissance de son mécanisme et peut parfois se coincer dans des optima locaux.

Titre	Auteurs	Année	Méthode d'optimisation	Avantages
Influence of Meta-heuristic Optimization on the Performance of Adaptive Interval Type2-fuzzy Traffic Signal Controllers	Araghi, S., Khosravi, A., Creighton, D., Nahavandi, S.	2016	SA, GA, CS	La recherche par cuckoo offre une convergence rapide mais risque une convergence prématurée. L'algorithme génétique est une alternative intéressante mais plus lente. Le recuit simulé garantit la découverte de l'optimum global mais a une convergence très lente.
Real-Time Detection of Vehicles for Advanced Traffic Signal Control	Han, C., Zhang, Q.	2008	Utilisation d'histogrammes de projection, d'un schéma de contrôle adaptatif des signaux, et d'une évaluation basée sur des paramètres de temps d'attente	Une gestion dynamique du trafic en fonction des conditions réelles et surveillance en temps réel, mais avec des défis liés à la précision de la détection et à la charge de traitement et la mise en œvre et à la disponibilité des données de circulation.
Simulation and Optimization of Traffic Lights For Vehicles Flow in High Traffic Areas	Ramirez-Polo, L., Jimenez-Barros, M. A., Varela Narváez, V., Parodi Daza, C	2022	Un modèle d'optimisation mis en œuvre à l'aide du programme Flexsim	Puissant pour optimiser précisément ce type de système, mais requiert des ressources de calcul. Les solutions trouvées sont spécifiques au cas d'étude.
State-of-art review of traffic signal control methods : challenges and opportunities	Qadri, S., Sultan, S., et al.	2020	PSO, GA, GP, et d'autres basées sur l'intelligence computationnelle	Efficace pour des problèmes complexes, capacité à échapper aux optima locaux, facilité de parallélisation. Mais temps de calcul parfois long, difficulté à prouver l'optimalité des solutions, nécessité d'un réglage fin des paramètres.

Titre	Auteurs	Année	Méthode d'optimisation	Avantages
Prédire la circulation à l'aide de l'apprentissage machine	Audet, O., O'Carroll, P.	2019	XGBoost	Gérer facilement de la circulation en utilisant des modèles d'apprentissage machine pour prédire les flux de trafic.
Evolutionary Algorithms for Optimizing Traffic Signal Operation	Vogel, A., Goerick, C., von Seelen, W.	2000	Hillclimbing	Hillclimbing présente des avantages en termes d'efficacité et de convergence rapide, mais il peut être limité par le risque de convergence locale et la sensibilité aux paramètres choisis.

TABLE 2.1 – Synthèse des travaux sur l'optimisation des feux de circulation.

2.5.8 Conclusion

Dans ce chapitre, nous avons présenté les méta-heuristiques, des techniques d'optimisation inspirées de la nature ou de phénomènes physiques, qui permettent de résoudre des problèmes complexes et combinatoires. Nous avons défini les méta-heuristiques, leurs intérêts, leurs avantages et leurs inconvénients.

Nous avons ensuite exposé les principales méta-heuristiques utilisées en optimisation, en distinguant les méta-heuristiques à solution unique, qui explorent l'espace de recherche à partir d'une seule solution initiale, et les méta-heuristiques à population de solutions, qui manipulent un ensemble de solutions simultanément. Nous avons décrit les principes et le fonctionnement de quelques exemples de méta-heuristiques, comme le recuit simulé, la recherche taboue, les algorithmes évolutionnaires, les colonies de fourmis et les essaims particulaires.

Enfin, nous avons passé en revue les applications des méta-heuristiques à l'optimisation du timing des feux de signalisation dans la littérature, en présentant les différentes approches et méthodes utilisées, ainsi que les résultats obtenus.

Chapitre 3

Approche proposée

3.1 Introduction

L'optimisation du temps de signalisation des feux de circulation est un défi majeur dans la gestion des réseaux routiers urbains. L'objectif principal est de minimiser les files d'attente et les temps de parcours, tout en assurant un écoulement fluide et sécurisé du trafic. Cependant, la complexité de ce problème d'optimisation combinatoire rend difficile l'obtention d'une solution exacte en un temps raisonnable, surtout lorsque le nombre d'intersections augmente.

Dans ce chapitre, une approche meta-heuristique hybride est proposée, combinant un algorithme génétique et un algorithme Snake Optimization. Cette approche vise à exploiter les forces complémentaires de ces deux techniques pour explorer efficacement l'espace de recherche et converger vers des solutions de haute qualité.

3.2 Description de la problématique

Le problème consiste à déterminer les temps de feu vert optimaux pour chaque direction (Nord, Sud, Est, Ouest) d'un carrefour routier, dans le but de minimiser la somme des files d'attente à toutes les approches. Les variables de décision sont les temps de feu vert du nord, sud, est et ouest pour les directions respectives. La fonction objective vise généralement à minimiser le temps d'attente des véhicules aux intersections.

Le problème est soumis à plusieurs contraintes. Premièrement, la somme des temps de feu vert de toutes les directions, additionnée aux temps de changement de phase, doit être égale au temps de cycle total du carrefour. Deuxièmement, les temps de feu vert de chaque direction doivent être compris entre des valeurs minimales et maximales, en fonction des exigences de sécurité et de fluidité du trafic. Enfin, les temps de feu vert doivent être des valeurs non négatives.

L'objectif est de coordonner les feux de signalisation de manière à permettre une circulation plus fluide, réduisant ainsi les congestions, les temps d'attente des véhicules aux intersections en ajustant les cycles de feux de signalisation et les durées des phases vertes.

La figure 3.1 [35] montre un exemple de l'intersection entre deux routes principales.

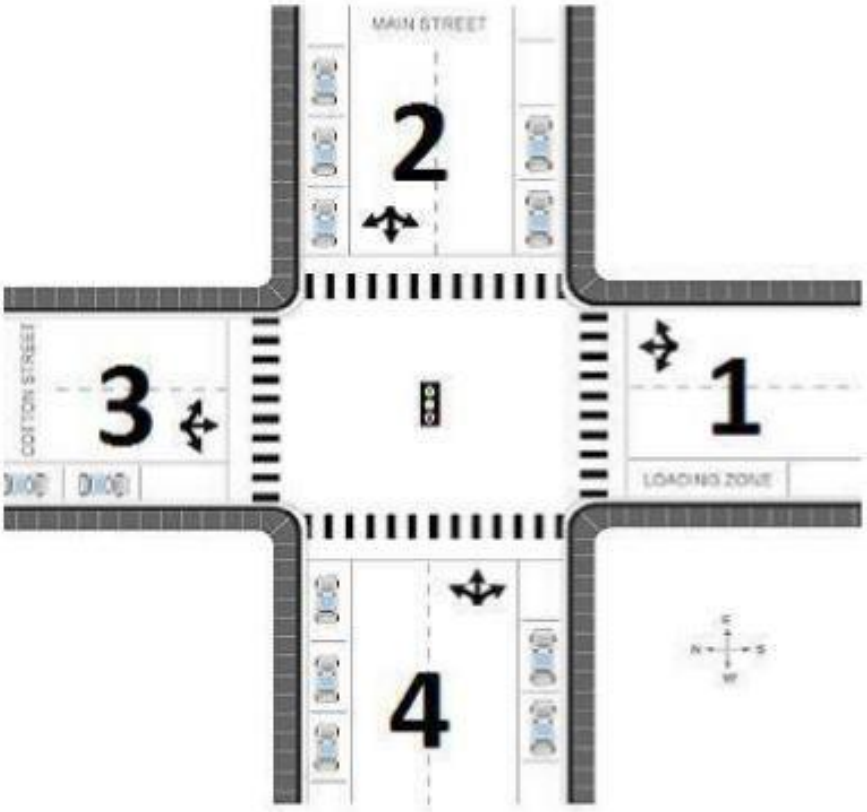

FIGURE 3.1 – Intersection entre deux routes principales

3.3 Architecture générale de l'approche proposée

L'architecture de l'approche proposée pour l'optimisation du temps de signalisation, combine une approche hybride associant un algorithme génétique (GA) et une optimisation par Snake Optimization (SO) avec le simulateur de trafic (SUMO).

Cette approche vise à identifier les paramètres de temps de signalisation optimaux pour minimiser la congestion et améliorer la fluidité du trafic. Une illustration schématique du processus est présentée dans la figure 3.2

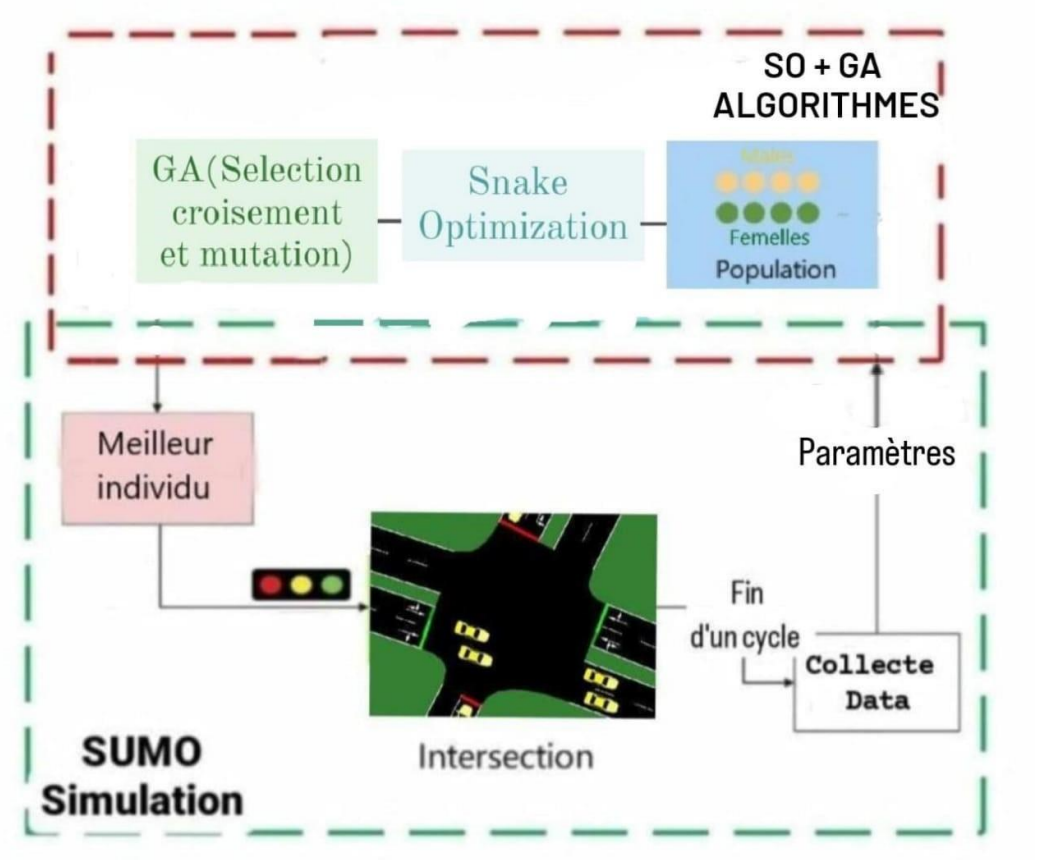

FIGURE 3.2 – Architecture générale de l'approche proposée

3.4 Description des Algorithmes proposés SO-GA

3.4.1 Algorithme de Snake Optimization (SO)

L'optimisation par l'algorithme Snake Optimization (SO) est une méthode d'optimisation intelligent proposé par Hashim et al.[40] basée sur le comportement des snakes pour résoudre des problèmes d'optimisation.

Cet algorithme s'inspire du mouvement des snakes lorsqu'ils cherchent leur proie, en utilisant des mécanismes de recherche pour trouver la meilleure solution à un problème donné.

Dans notre approche proposé, l'algorithme du SO est intégré à l'algorithme génétique pour améliorer l'exploration de l'espace de recherche et éviter les optimums locaux.

Il représente une méthode complémentaire pour améliorer les solutions découvertes par l'algorithme génétique. Dans ce processus, une métaphore est employée

où un "snake" représente une solution candidate, se déplaçant dans l'espace de recherche en ajustant les valeurs du plan de feux à chaque itération. Cette métaphore est guidée par une fonction d'évaluation qui oriente le mouvement du snake vers des solutions présentant un meilleur fitness.

Initialisation

Création des populations initiale

Dans notre cas, la population initiale sera composée d'un certain nombre de snakes créé chacun en permutant aléatoirement les gènes du chromosome initial codant une attribution initiale des temps de feux verts des différentes directions de l'intersection.

L'algorithme suivant résume les étapes de la création de la population initiale :

Création de la population

```
Début
Population initiale = Ø
Répéter
1. Permuter aléatoirement les gènes du chromosome
initial
2. Ajouter l'individu résultant à la population
initiale.
Jusqu'à ce que le nombre d'individus spécifié soit
atteint.
Fin
```

FIGURE 3.3 – Création de la population initiale

La population est ensuite divisée de manière égale en deux groupes, mâles et les femelles selon les equations (1) et (2)

Où :

$$N_{female} \cong \frac{N}{2} \tag{3.1}$$

$$N_{male} = N - N_{female} \tag{3.2}$$

Les principales phases de l'algorithme SO [41] :

Dans l'algorithme, la température (T) et la quantité de nourriture (Q) sont calculées selon les équations (3) et (4) :

$$T_{emprature} = \exp^{(\frac{-t}{T})} \tag{3.3}$$

Où t fait référence à l'itération actuelle, et T est le nombre maximum d'itérations.

$$Q = 0.5 \times exp^{(\frac{(t-T)}{T})} \tag{3.4}$$

Phase exploration :

Une valeur de $(Q < 0.25)$ indique une nourriture insuffisante dans l'environnement, ce qui signifie que les snakes sont en phase d'exploration et cherchent de la nourriture de manière aléatoire.

Les individus (snakes) explorent l'espace de recherche à la recherche de nourriture, en mettant à jour leurs positions (temps de feu vert) en fonction de leur capacité à trouver de la nourriture et de la quantité de nourriture disponible.

La phase d'exploration est exprimée comme suit dans les equations (5) et (6) :

$$X_{t+1,i,m} = X_{t,random,m} \pm C_2 \times A_m \times ((X_{max} - X_{min}) \times rand + X_{min}) \tag{3.5}$$

$$X_{t+1,i,f} = X_{t,random,f} \pm C_2 \times A_f \times ((X_{max} - X_{min}) \times rand + X_{min}) \tag{3.6}$$

Où :

— $X_{t+1,i,m}$ et $X_{t+1,i,f}$ sont les positions du i-ème mâle et femelle respectivement.

— $X_{t,random,m}$ et $X_{t,random,f}$ sont les positions d'individus sélectionnés aléatoirement parmi la population mâle ou femelle.

— C_2 est un constante.

— A_m et A_f représentent la capacité des mâles et des femelles à trouver de la nourriture.

Calculés avec les equations (7) et (8) :

$$A_m = exp^{(-\frac{f_{(rand,m)}}{f_{i,m}})} \tag{3.7}$$

$$A_f = exp^{(-\frac{f_{(rand,f)}}{f_{i,f}})} \tag{3.8}$$

— $f_{rand,m}$ et $f_{rand,f}$ sont les valeurs de fitness des individus sélectionnés aléatoirement parmi les mâles et les femelles.

— $f_{i,m}$ et $f_{i,f}$ sont les valeurs de fitness du i-ème individu dans les groupes des mâles et des femelles respectivement.

— rand est une valeur aléatoire entre 0 et 1.

— ($X_{max}X_{min}$) définissent les bornes des positions.

Phase d'exploitation :

Dans la phase d'exploitation, il y a suffisamment de nourriture disponible dans l'environnement. Si la température est supérieure à 0,6, les snakes continuent à chercher de la nourriture. Les positions des mâles et des femelles peuvent être mises à jour selon l'équation (9)

$$X_{t+1,i,j} = X_{food} \pm C_3 \times Temp \times rand \times (X_{food} - X_{t,i,j}) \tag{3.9}$$

Où $X_{i,j}$ est la nouvelle position d'un individu dans les populations mâle et femelle.

Mode combat et accouplement :

Si la température est inférieure à 0,6, les snakes entrent dans un état de combat ou d'accouplement.

Mode combat (fighting) :

Selon les équations (10) et (11) :

$$X_{t+1,i,m} = X_{t,i,m} \pm C_3 \times FM \times \text{"}rand\text{"} \times (X_{\text{"}best\text{"},f} - X_{t,i,m}) \tag{3.10}$$

$$X_{t+1,i,f} = X_{t+1,i,f} \pm C_3 \times FF \times \text{"}rand\text{"} \times (X_{\text{"}best\text{"},m} - X_{t+1,i,f}) \tag{3.11}$$

$$FM = exp^{(-\frac{f_{\text{"}best\text{"},f}}{f_i})} \tag{3.12}$$

$$FF = exp^{(-\frac{f_{\text{"}best\text{"},f}}{f_m})} \tag{3.13}$$

Où FM et FF représentent respectivement les capacités de combat des mâles et des femelles, et $X_{best,m}$ et $X_{best,f}$ sont les positions des meilleurs individus mâles et femelles, respectivement. $f_{best,f}$ et $f_{best,m}$ indiquent l'aptitude de $X_{best,f}$ et $X_{best,m}$, respectivement.

Mode d'accouplement (mating) :

Selon les équations (14) et (15)

$$X_{t+1,i,m} = X_{t,i,m} \pm C_3 \times M_m \times \text{"}rand\text{"} \times (Q \times X_{t,i,f} - X_{t,i,m}) \tag{3.14}$$

$$X_{t+1,i,f} = X_{t,i,f} \pm C_3 \times M_f \times \text{"}rand\text{"} \times (Q \times X_{t,i,m} - X_{t,i,f}) \tag{3.15}$$

$$MM = exp^{(-\frac{f_{i,f}}{f_{i,m}})} \tag{3.16}$$

$$MF = exp^{(-\frac{f_{i,m}}{f_{i,f}})} \tag{3.17}$$

Où MM et MF représentent respectivement les capacités d'accouplement des mâles et des femelles.

C_3 est une constante.

L'algorithme du Snake optimization est appliqué à chaque génération. Les paramètres tels que la température et la quantité de nourriture sont mis à jour en fonction de la génération actuelle, et les différentes phases de l'algorithme de Snake sont appliquées en conséquence.

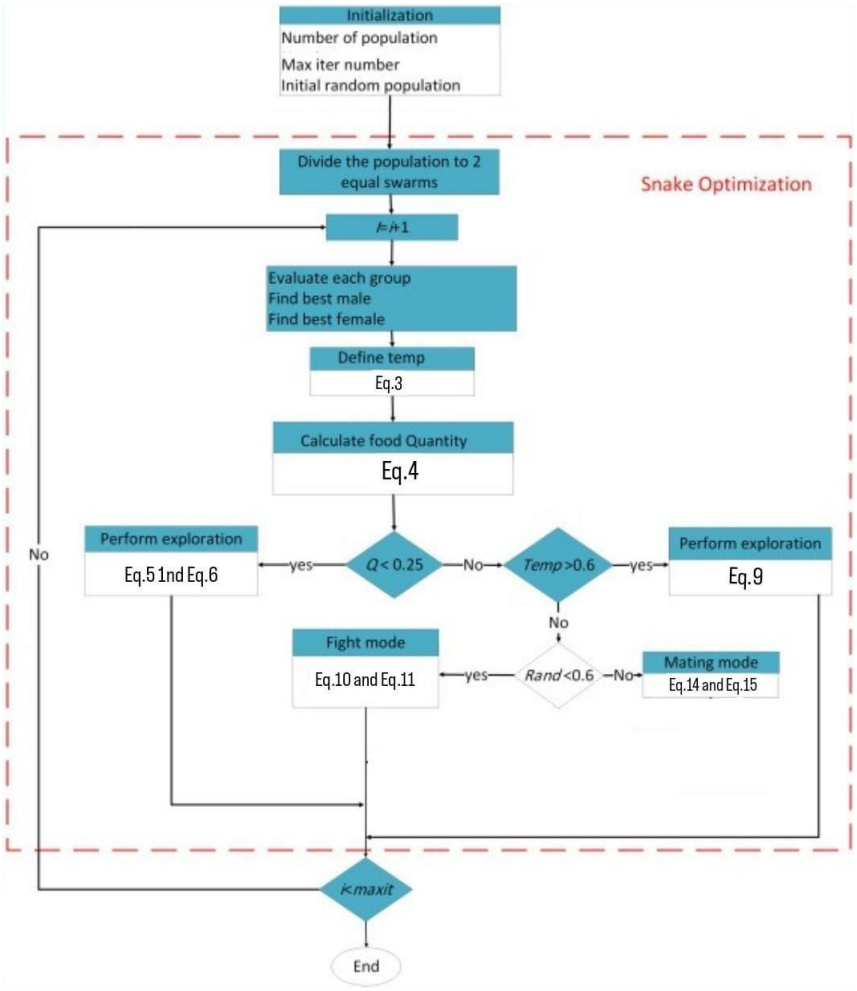

FIGURE 3.4 – Organigramme de Snake Optimization

3.4.2 Optimisation par algorithme génétique

Un algorithme génétique est un type d'algorithme évolutionniste qui utilise des techniques inspirées de la biologie évolutive telles que l'hérédité, la mutation, la sélection naturelle, et le croisement (ou recombinaison) pour générer des solutions de haute qualité à des problèmes d'optimisation et de recherche. Développés dans les années 70 par John H. Holland [35].

Les algorithmes génétiques (GA) standards commencent par transformer les paramètres d'un problème d'optimisation en une séquence codée de longueur déterminée. L'idée est d'imiter le processus évolutif naturel au sein d'une population de candidats jusqu'à atteindre un point d'arrêt prédéfini. On initie ce processus en

créant une population de départ composée de solutions potentielles.

À chaque génération, certains candidats sont choisis en fonction de leur performance évaluée par une fonction objective(adaptation). Ensuite, on applique des opérations de croisement et de mutation pour générer de nouveaux individus, renouvelant ainsi la population.

Cette procédure se répète jusqu'à atteindre un critère d'arrêt spécifique, souvent le nombre maximal de générations souhaitées.

Ce mécanisme est illustré dans la figure 3.3 [36] comme un exemple de l'approche standard des GA.

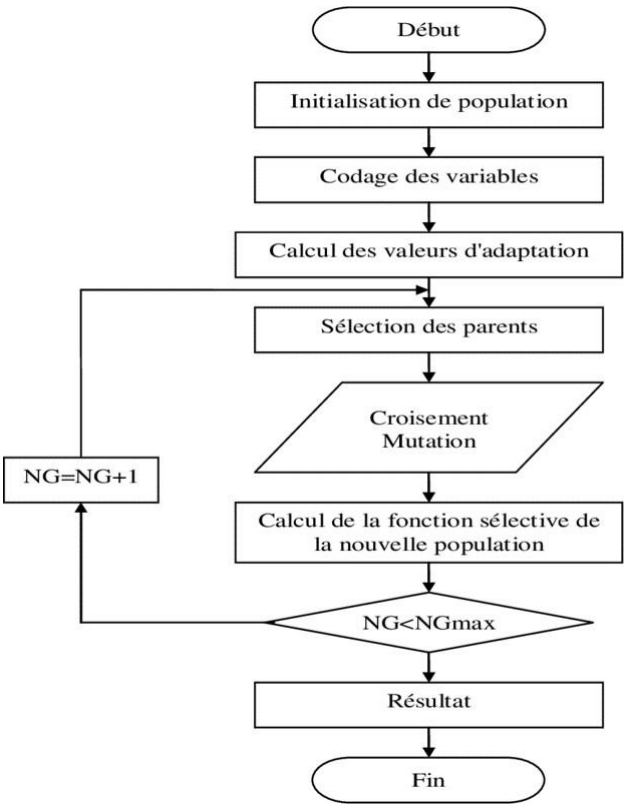

FIGURE 3.5 – Organigramme générale d'un algorithme génétique[36].

L'ensemble des étapes suivantes [37] présente une lecture du contenu de l'organigramme de la figure 3.3.

— **Etape 0 :** Définir un codage du problème
— **Etape 1 :** Créer une population initiale p_0 de q individus $x_1, x_2, ., x_q i = 0$;
— **Etape 2 :** Evaluation des individus. Soit F la fonction d'évaluation. Calculer $F(x_i)$ pour chaque individu x_i de p_i

— **Etape 3 :** Sélection Sélectionner les meilleurs individus (au sens de F) et les grouper par paire.

— **Etape 4 :** Application des opérateurs génétiques

1. **Croisement :** appliquer l'opération de croisement aux paires sélectionnées
2. **Mutation :** appliquer la mutation aux individus issus du croisement

Ranger les nouveaux individus obtenus (de 1 et 2) dans une nouvelle génération p_{i+1},

Répéter les étapes 2, 3 et 4 jusqu'à l'obtention du niveau de performance souhaité

3.4.3 Le codage

L'étape de codage est l'une des étapes les plus important de l'algorithme génétique d'un chromosome. Une solution candidate (chromosome) dans note cas est un vecteur de 4 nombres aléatoires (gènes). Chaque gène correspondant à un temps de feu vert est constitué d'un nombre aléatoire, entre 15 et 55 secondes.

Réellement, c'est le codage decimal qui est considéré pour permettre la génération (par application des opérateurs de reproduction) de nouvelles valeurs de gènes (temps verts) permettant de s'adapter aux changements réels de circulation sur la route.

En considérant une intersection ayant 4 phases avec les temps de feu vert pour les directions Nord/Sud doivent être identiques, et les temps de feu vert pour les directions Est/Ouest doivent également être identiques. comme l'exemple illustré sur la figure ci-dessous.

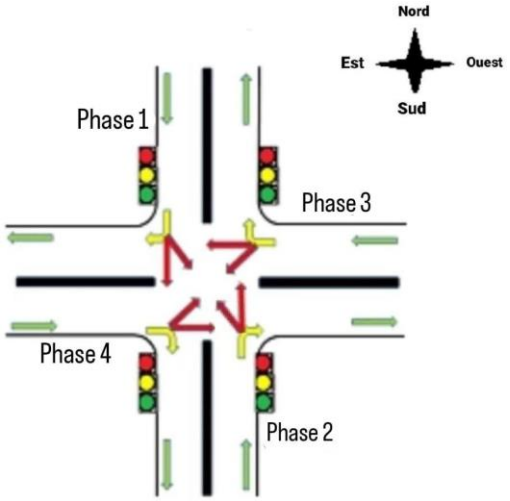

FIGURE 3.6 – Exemple des phases sur une intersection contrôlée par des feux de signalisation.

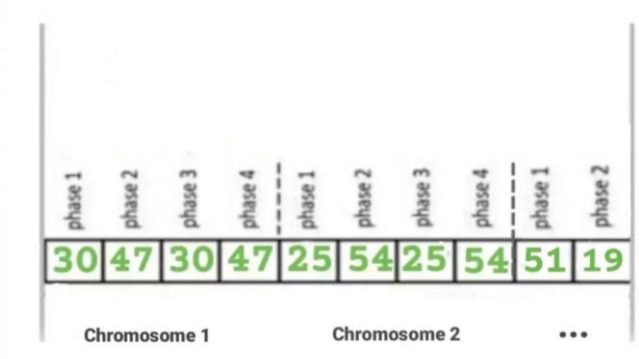

FIGURE 3.7 – Codage d'un chromosome pour contrôler deux feux à 4 phases respectivement.

L'évaluation des individus

La fonction d'évaluation quantifie la qualité de chaque chromosome par rapport au problème. Les chromosomes ayant une bonne qualité ont plus de chance d'être sélectionnés pour la reproduction, et donc plus de chance pour survivre d'une génération à une autre.

La fonction d'adaptation produit la pression qui permet de faire évoluer la population de l'algorithme évolutionnaire vers les individus de meilleure qualité.

En clair, le choix de la fonction d'évaluation va fortement influer sur le succès de l'algorithme.

La sélection des individus

Après l'évaluation, les individus sont maintenant prêts à passer par l'étape de sélection Le rôle de la sélection est de distinguer entre les individus sur la base de leur qualité, en particulier, pour permettre aux meilleurs individus de devenir parents dans la génération suivante. Ainsi, elle est responsable sur le fait de pousser l'amélioration de la qualité.

Dans notre cas l'opérateur utilisé est une sélection de type tournoi pour choisir les individus qui vont se reproduire et former la prochaine génération.

La méthode de création et d'évolution des populations dans les tournois génétiques se divise en plusieurs étapes. Tout d'abord, la population est aléatoirement divisée en plusieurs groupes, appelés tournois, chacun comprenant un nombre fixe d'individus (5).

Ensuite, pour chaque tournoi, les individus sont évalués selon leur "fitness", qui mesure leur qualité par rapport à un critère spécifique. Les meilleurs individus sont sélectionnés, en fonction de la stratégie adoptée, pouvant aller d'un seul gagnant à plusieurs.

Les individus ainsi choisis sont ensuite reproduits dans une nouvelle population de même taille. Si nécessaire, d'autres tournois sont organisés pour compléter la nouvelle population.

Sélection par tournoi

```
Début
    Population = [liste des individus dans la population]
    Taille_tournoi = [spécifiez la taille du tournoi]
    Nombre_sélectionné = [spécifiez le nombre d'individus à sélectionner]
    Population_sélectionnée = Ø
    Répéter Nombre_sélectionné fois
      1.  Evaluer_fitness = [Évaluez le fitness de chaque individu
sélectionné]
      2.  Individus_au_tournoi = [sélectionner aléatoirement Taille_tournoi
individus de la population]
      3.  Meilleur_individu = [trouver l'individu avec le meilleur fitness
parmi Individus_au_tournoi]
      4.  Ajouter Meilleur_individu à Population_sélectionnée
    Fin Répéter
    Retourner Population sélectionnée
```

FIGURE 3.8 – Algorithme de sélection par tournoi

Enfin, sur cette nouvelle population, des opérateurs génétiques tels que le croisement et la mutation sont appliqués pour générer des descendants. Ces descendants sont ajoutés à la population.

3.4.4 Opérateurs de reproduction

Croisement

Le croisement, également connu sous le nom de recombinaison, est un autre opérateur clé utilisé dans les algorithmes génétiques pour créer de nouvelles solutions en combinant les caractéristiques des individus sélectionnés pour la reproduction.

Le croisement implique la création de nouveaux individus en combinant les informations génétiques de deux parents pour produire une descendance. Cela se fait en échangeant des parties des gènes des parents pour créer des enfants qui héritent des caractéristiques de leurs parents.
L'algorithme suivant illustre les étapes du croisement.

Croisement

```
Début
  1.  Sélection de deux individus au hasard.
  2.  Génération de deux points de croisement au hasard.
  3.  Application de l'opérateur de croisement.
  4.  Ajoute des enfants à la population.
Fin
```

FIGURE 3.9 – Algorithme de croisement.

La figure suivante donne un exemple applicatif du croisement à un point.

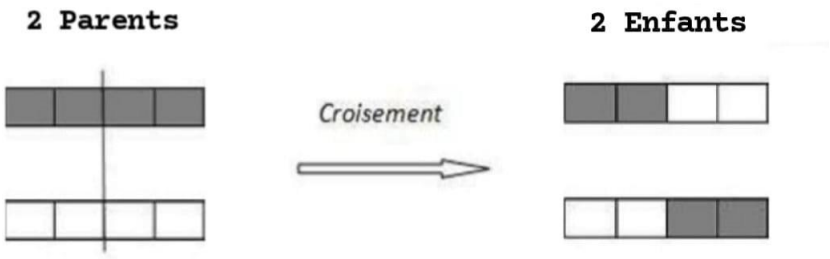

FIGURE 3.10 – Exemple du croisement à un point

Mutation

La mutation est l'un des opérateurs clés utilisés dans les algorithmes génétiques pour introduire de la diversité génétique au sein de la population d'individus. L'objectif principal de la mutation est d'introduire de nouvelles informations génétiques qui ne sont pas présentes dans la population actuelle, ce qui permet d'explorer de nouvelles régions de l'espace de recherche et d'éviter la convergence prématurée vers des solutions sous-optimales.

L'opérateur de mutation manipule les gènes d'un individu choisi au hasard. Dans ce cas, la mutation sera appliquée à l'individu en effectuant une permutation entre deux gènes de l'individu. L'individu résultant sera ajouté à la population.

L'algorithme suivant montre les étapes de l'opérateur de mutation :

Mutation

```
Début
  1.  Sélectionner un individu aléatoirement
  2.  Choisir deux positions i et j tel que 0 ≤ i, j ≤ taille de
l'individu et i ≠ j.
  3.  Permuter entre le gène i et le gène j de l'individu pour
produire un individu fils.
  4.  Ajouter le fils à la population.
Fin
```

FIGURE 3.11 – Algorithme de mutation

3.4.5 Gestion des contraintes

Un élément de population qui viole une contrainte se verra attribuer un mauvais 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. [39]

3.4.6 Critères d'arrêt

Après l'exécution des étapes précédentes (croisement, mutation, évaluation, sélection) pour un nombre spécifique d'itération représentant le nombre de génération produites, l'exécution sera terminée lorsque le nombre d'itération atteindra le nombre spécifié.

3.4.7 Hybridation de SO et GA

L'approche proposée hybride l'algorithme snake optimization (SO) avec l'algorithme génétique (GA) dans le but de tirer parti des forces complémentaires de ces deux techniques méta-heuristiques.

Le GA est reconnu pour son efficacité à explorer globalement l'espace de recherche grâce à ses opérateurs de sélection, croisement et mutation inspirés des principes de l'évolution naturelle. Cependant, il peut parfois stagner dans des optima locaux et avoir des difficultés à affiner davantage les solutions prometteuses.

C'est là qu'intervient snake optimization, une technique d'optimisation locale inspirée du comportement des snakes. Elle consiste à représenter une solution sous forme d'un "snake" et à effectuer des mouvements locaux prédéfinis pour améliorer progressivement cette solution en explorant son voisinage immédiat.

En intégrant le SO au sein de GA, nous visons à compléter ce dernier en affinant les solutions générées par les opérateurs génétiques.

Dans cette approche hybride, GA et SO interagissent et se complètent mutuellement de la manière suivante. À chaque génération, l'algorithme de snake optimization (SO) est appliqué à la population afin d'améliorer localement la solution. La nouvelle solution optimisée par le SO remplace alors l'individu d'origine dans la population si elle présente une meilleure qualité (fitness).

Des parents sont sélectionnés à partir de la population, ils sont croisés pour générer de nouveaux enfants. Les enfants subissent une mutation avec une probabilité donnée et ne nouvelle population est générée à partir des enfants obtenus. Si la condition d'arrêt est satisfaite, la meilleure solution (temps de feu vert optimaux et fitness correspondante) est affichée. Sinon, l'algorithme passe à la génération suivante.

Les solutions améliorées localement par SO sont réinjectées dans la population de GA, offrant ainsi de meilleurs points de départ pour les générations futures. Cette interaction bidirectionnelle entre l'exploration globale de le GA et l'optimisation locale de le SO permet d'accélérer la convergence vers des solutions de haute qualité tout en évitant de rester bloqué dans des optima locaux.

3.4.8 Fonction de fitness

Dans le cadre du contrôle des feux de circulation, le retard a un impact significatif sur l'évaluation du flux de circulation actuel et est donc souvent utilisé comme indicateur clé De l'efficacité de la circulation.

La formule de Webster et amélioré par Yang [42], largement utilisée dans ce domaine, calcule le délai comme suit :

$$D_i = \frac{C_q(x_i - y)^2}{2x(1-y)} + \frac{x^2}{2(1-x)} \tag{3.18}$$

Où C est le temps d'un cycle, q_i est le débit de trafic correspondant de la phase i, et yi est la saturation de la phase i et x est la saturation de l'intersection.

Afin de disposer d'un critère raisonnable pour le plan de synchronisation, le temps de retard moyen des véhicules du cycle est utilisé comme indice d'évaluation du plan de synchronisation des signaux; la minimisation de la formule est donc l'objectif de la fonction d'optimisation [29] :

$$minD = \frac{1}{n} \sum_{i=1}^{n} \frac{D_i}{N_i} \tag{3.19}$$

Où n est le nombre de voies, D_i est le temps de retard du cycle de la i-ème voie, N_i est le flux de cycle de chaque voie, et D est le temps de retard moyen des véhicules du cycle.

Symbole	Description	Unité
C	Temps d'un cycle	S
D_i	Temps de retard par cycle de chaque phase	S
q_i	Débit de trafic de la phase i	(Veh/h)
n	Nombres de voies	
N_i	Débit de trafic des voies dans la phase i	
y_i	Saturation de trafic de la phase i	
X	Saturation de l'intersection	

TABLE 3.1 – Les paramètres associés dans fonction de fitness

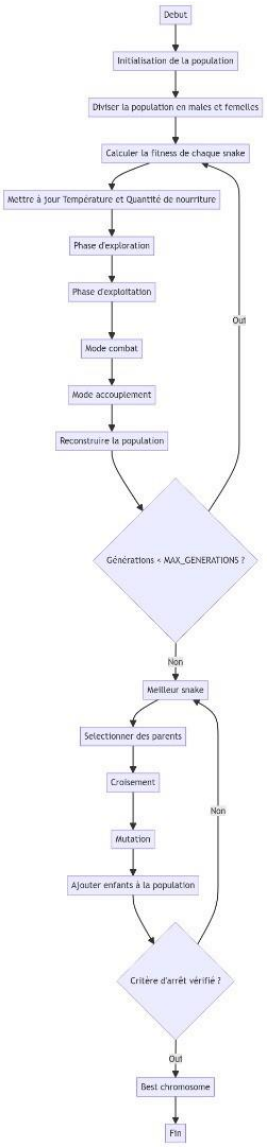

FIGURE 3.12 – Organigramme de l'approche proposée

3.5 Conclusion

En conclusion, ce chapitre a exploré l'hybridation de l'algorithme génétique avec l'optimisation par snake comme une méthode prometteuse pour l'optimisation des feux de circulation.

L'approche proposée capitalise sur la robustesse de l'algorithme génétique et la précision de l'optimisation par snake pour former une stratégie d'optimisation puissante et flexible.

La fonction de fitness développée offre un moyen efficace d'évaluer et de comparer les différentes configurations de feux, en tenant compte de multiples critères tels que les files d'attente et les délais.

Chapitre 4

Résultats expérimentaux

4.1 Introduction

L'objectif principal de ce projet est d'optimiser la programmation des feux de signalisation dans une intersection routière afin de minimiser le temps d'attente des véhicules et, par conséquent, d'améliorer la fluidité du trafic.

Pour atteindre cet objectif, deux approches meta-heuristiques ont été explorées : les algorithmes génétiques et snake optimization (SO).

Dans ce chapitre, nous présentons et analysons les résultats expérimentaux obtenus en appliquant ces méthodes d'optimisation sur un environnement de simulation réaliste. Les algorithmes GA et SO ont été implémentés en Java et Python, tandis que le simulateur de trafic urbain SUMO (Simulation of Urban MObility) a été utilisé pour modéliser une intersection routière et évaluer les performances des solutions proposées.

4.2 L'environnement de développement :

4.2.1 Python 3.9 :

Python est un langage de programmation de haut niveau développé par Guido Van Rossum et de nombreux contributeurs bénévoles. Il est un langage portable, dynamique, extensible, gratuit qui permet une approche modulaire et orienté objet de la programmation.

Il est reconnu pour sa syntaxe intuitive et sa facilité d'écriture, ce qui accélère le développement et le prototypage.

Python est doté d'une riche collection de bibliothèques scientifiques et de simulation, qui sont particulièrement utiles pour la modélisation, l'analyse de données et la visualisation dans le cadre de simulations [42].

4.2.2 Java :

Avec sa machine virtuelle Java (JVM), il assure la portabilité et la performance à travers différentes plateformes, ce qui est crucial pour les simulations complexes qui doivent être exécutées dans divers environnements sans modification du code source.

La capacité de Java à gérer efficacement la mémoire et à exécuter des processus en parallèle grâce à son modèle de multithreading en fait un choix solide pour les simulations nécessitant des calculs intensifs et une exécution concurrente [44].

Outils de Java :

Le **langage de programmation Java** est accompagné de plusieurs outils et composants utilisés pour le développement, l'exécution et la gestion des applications Java.

Voici quelques-uns de ces éléments :

— **Java Development Kit (JDK) :** Le JDK est un kit de développement logiciel qui inclut tous les outils nécessaires pour développer et exécuter des applications Java, tels qu'un compilateur, un débogueur et un environnement d'exécution.

— **Java Runtime Environment (JRE) :** Le JRE est un ensemble de logiciels qui fournit l'environnement d'exécution nécessaire pour exécuter des applications Java.

— **Java Virtual Machine (JVM) :** La JVM est un composant logiciel qui offre un environnement d'exécution pour les applications Java. Elle traduit le bytecode Java en instructions natives du processeur et permet l'exécution indirecte de programmes sur différents systèmes d'exploitation ou plates-formes.

— **Software Development Kit (SDK) :** Le SDK est une collection d'outils de développement logiciel utilisés pour créer, tester et déployer des applications.

4.2.3 Le simulateur SUMO

SUMO est l'acronyme de "Simulation of Urban MObility". C'est un progiciel de simulations de trafic routier open source sous licence ONU public (OPL), mis en Suivre en 2001, avec une première version open source en 2002.

L'objectif des développeurs est de mettre à la disposition du monde académique un outil leur permettant de modéliser le réseau routier aussi bien en milieu urbain que sur les autoroutes ou les rocades (contrairement à ce qui est indiqué sur son appellation). SUMO est utilisé dans le cadre de plusieurs projets de recherche nationaux et internationaux.

Les contextes des études sont très variés tels : Évaluation des feux de circulation, Choix d'itinéraire et re-routage, Évaluation des méthodes de surveillance du trafic, Simulation de communications véhiculaires, Prévisions de trafic. [44]

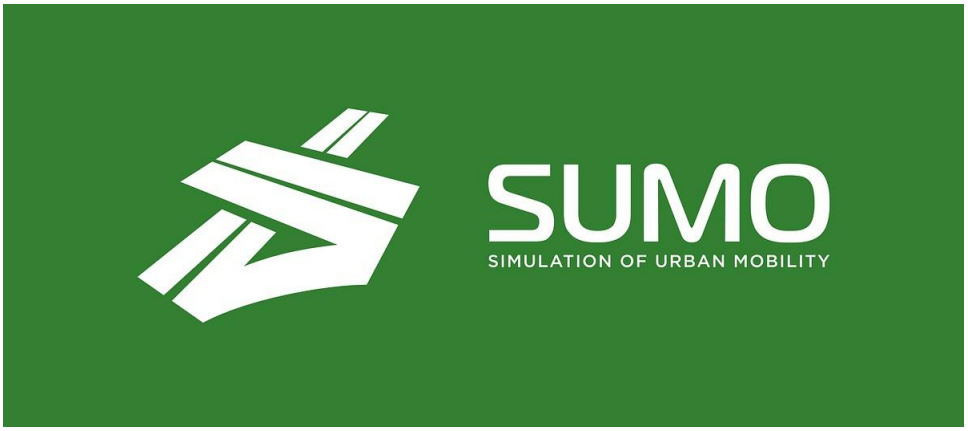

FIGURE 4.1 – SUMO

Principales fonctionnalités

SUMO n'est pas seulement une simulation de trafic, mais plutôt une suite d'applications qui aident à préparer et à exécuter les simulations de trafic.

Comme la simulation de trafic en SUMO nécessite la représentation des réseaux routiers et de la demande du trafic dans un format précis, les deux doivent être importés ou générés à l'aide de différentes sources.

Les réseaux routiers SUMO peuvent être soit générés à l'aide d'une application nommée «netgenerate» ou générée par l'importation d'une carte de route numérique.

Applications du paquet SUMO

SUMO est un logiciel complet qui comprend plusieurs applications nécessaires à la préparation de la simulation :

— SUMO : La simulation microscopique sans visualisation; application en ligne de commande.

— SUMO-GUI : La simulation microscopique avec une interface utilisateur graphique.

— NETCONVERT : Possède la capacité d'importer et générer des cartographies adaptées pour SUMO à partir de différents formats.

— NETGENERATE : Permet de générer en ligne de commande un réseau de plusieurs formes : en grille, en toile ou aléatoire.

— NETEDIT : Un éditeur de réseau graphique.

4.3 Étude de la régulation du trafic à l'intersection du Village Boudia

Pour tester les performances de notre approche visant à réguler le trafic routier et à optimiser les temps d'attente des véhicules, nous avons analysé le réseau routier en tenant compte du flux des véhicules. Nous avons choisi de travailler sur une intersection de la ville de Saida, en nous basant sur la quantité de flux quotidien à cet endroit.

Les feux de signalisation routière dans cette zone fonctionnent selon un système rudimentaire. Lorsque le feu rouge s'allume, les voitures doivent s'arrêter pendant toute la durée du feu, qui est de 27 secondes, même si la voie opposée est libre. Cela favorise l'immobilité des véhicules, générant de nombreux problèmes : embouteillages, perte de temps et de carburant pour les automobilistes, pollution par les gaz d'échappement, stress et nuisances sonores.

En revanche, la durée du feu vert, synonyme de mobilité, est de 35 secondes. Il est donc évident que le système en place n'est plus adapté à la perspective actuelle, qui privilégie la fluidification de la circulation routière par la réduction des temps d'arrêt et l'optimisation du fonctionnement des feux de signalisation.

Les figures suivantes montrent des captures en OpenStreetMap sur l'intersection choisit.

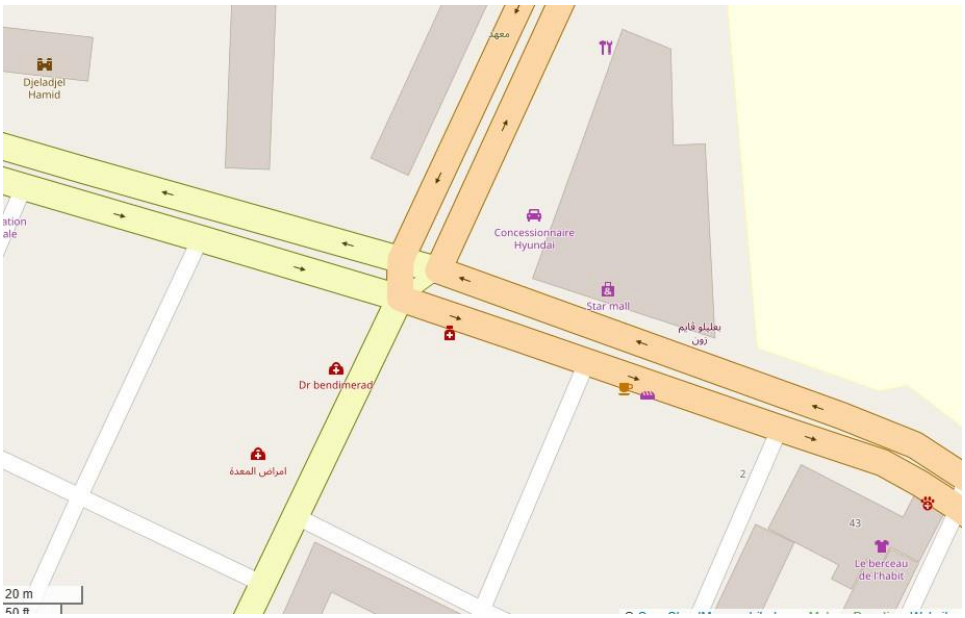

FIGURE 4.2 – Vue de l'intersection de village Boudia Saida en OpenStreeMap

4.3.1 Préparation de scénario

Le réseau est importé à partir de osmWebWizard et converti en un réseau SUMO (osm.net.xml) au moyen d'une série de scripts fournis dans le package SUMO, la figure 4.5 montre le detail de ce fichier. Nous avons choisi osmWebWizard car il permet de générer des scénarios réels pour la simulation de la mobilité urbaine avec SUMO.

osmWebWizard commence par télécharger les données OpenStreetMap pour la zone géographique sélectionnée. Ces données incluent les informations sur les routes, les intersections et les points d'intérêt.

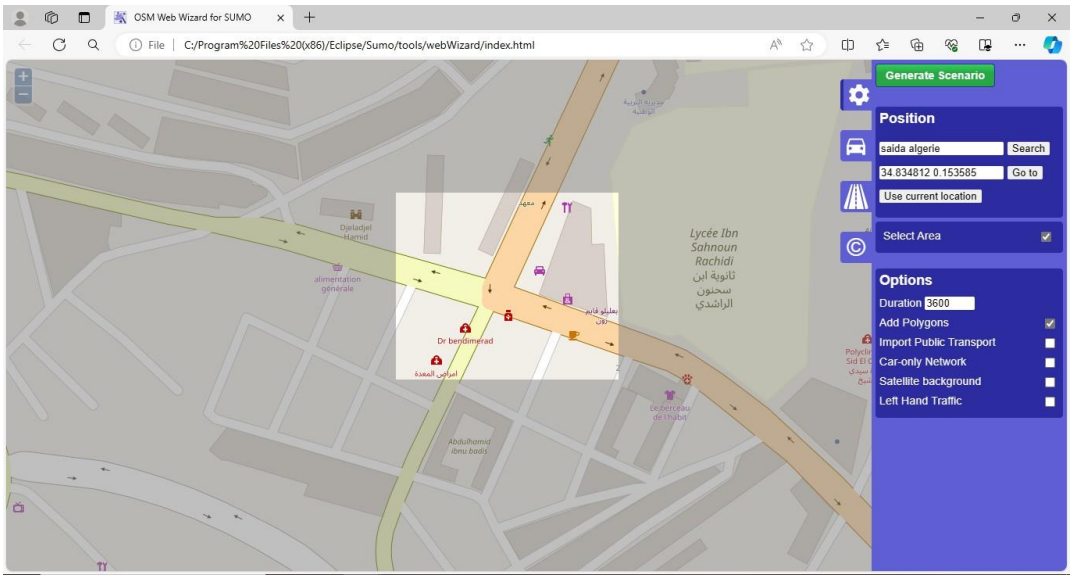

FIGURE 4.3 – l'intersection étudier en osmWebWizard

On Clique sur le bouton "Generate Scenario" pour générer le scénario. Une fois le scénario généré, osmWebWizard ouvrira automatiquement l'interface graphique SUMO-GUI (osm.sumocfg) présente dans la figure 4.4

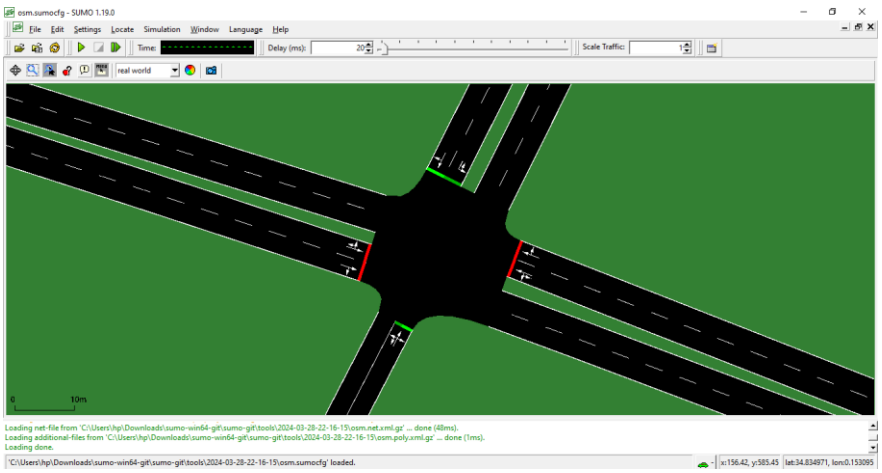

FIGURE 4.4 – Réseau généré dans SUMO

```
<net xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.16" junctionCornerDetail="5" limitTurnSpeed="5.50"
xsi:noNamespaceSchemaLocation="http://sumo.dlr.de/xsd/net_file.xsd">
  <location netOffset="-239279.57,-3857853.13" convBoundary="0.00,0.00,1795.74,948.11" origBoundary="0.148858,34.829819,0.168623,34.838147" projParameter="+proj=utm +zone=31
  +ellps=WGS84 +datum=WGS84 +units=m +no_defs"/>
  <type id="highway.primary" priority="12" numLanes="2" speed="27.78" disallow="tram rail_urban rail rail_electric rail_fast ship" oneway="0"/>
  <type id="highway.residential" priority="3" numLanes="1" speed="13.89" disallow="tram rail_urban rail rail_electric rail_fast ship" oneway="0"/>
  <type id="highway.secondary" priority="11" numLanes="1" speed="27.78" disallow="tram rail_urban rail rail_electric rail_fast ship" oneway="0"/>
  <type id="highway.tertiary" priority="10" numLanes="1" speed="22.22" disallow="tram rail_urban rail rail_electric rail_fast ship" oneway="0"/>
  <edge id=":1829318971_0" function="internal">
    <lane id=":1829318971_0_0" index="0" disallow="tram rail_urban rail rail_electric rail_fast ship" speed="11.11" length="11.75" shape="82.47,474.76 82.39,471.52 82.13,468.90
    81.62,466.31 80.79,463.18"/>
  </edge>
  <edge id=":1829318971_1" function="internal">
    <lane id=":1829318971_1_0" index="0" disallow="tram rail_urban rail rail_electric rail_fast ship" speed="7.90" length="3.45" shape="82.47,474.76 82.94,471.48 83.01,471.36"/>
  </edge>
  <edge id=":1829318971_2" function="internal">
    <lane id=":1829318971_2_0" index="0" disallow="tram rail_urban rail rail_electric rail_fast ship" speed="3.65" length="1.44" shape="82.47,474.76 83.26,473.56"/>
  </edge>
  <edge id=":1829318971_9" function="internal">
    <lane id=":1829318971_9_0" index="0" disallow="tram rail_urban rail rail_electric rail_fast ship" speed="7.90" length="9.23" shape="83.01,471.36 84.41,469.01 86.86,467.36
    90.30,466.52"/>
  </edge>
  <edge id=":1829318971_10" function="internal">
    <lane id=":1829318971_10_0" index="0" disallow="tram rail_urban rail rail_electric rail_fast ship" speed="3.65" length="3.23" shape="83.26,473.56 84.06,473.15 84.86,473.55
    85.67,474.75"/>
  </edge>
  <edge id=":1829318971_3" function="internal">
    <lane id=":1829318971_3_0" index="0" disallow="tram rail_urban rail rail_electric rail_fast ship" speed="6.37" length="7.91" shape="90.65,469.70 88.46,470.22 86.90,471.24
    85.97,472.75 85.67,474.75"/>
```


FIGURE 4.5 – Le fichier osm.net.xml

4.3.2 Interaction avec la simulation

Contrôle des simulations via TRACI

SUMO offre également une interface nommée "TraCI(Traffic Control Interface)" qui permet de connecter le simulateur avec d'autre applications externes pour la supervision, la récupération des résultats et la modification de simulation en cours d'exécution.

Utilisation TRACI en Python pour collecter les données

Afin de capturer la dynamique complexe du trafic à notre intersection étudie, nous avons développé un script Python qui s'intègre parfaitement à l'environnement de simulation de trafic SUMO via l'interface TraCI. Ce script permet d'extraire des paramètres détaillés des véhicules et de les stocker dans un fichier vehicle_flow_data facilement accessible.

Le processus débute par l'importation des bibliothèques requises, y compris traci pour l'interaction avec SUMO et csv pour la manupilation des fichiers. On lance ensuite la simulation SUMO via le fichier de configuration et on établit une liaison avec TraCI. Une boucle de simulation est mise en place pour simuler les étapes de temps souhaitées.

À chaque pas de temps, il détermine la phase actuelle des feux pour chaque intersection, puis collecte des métriques clés sur le flux de véhicules pour chaque voie contrôlée par ces feux, notamment le nombre de véhicules, leur vitesse moyenne, le débit, le taux de débit et la saturation. Toutes ces données sont enregistrées dans le fichier.

L'exécution du programme Python démarre SUMO et lance la simulation présentée par les figures suivantes :

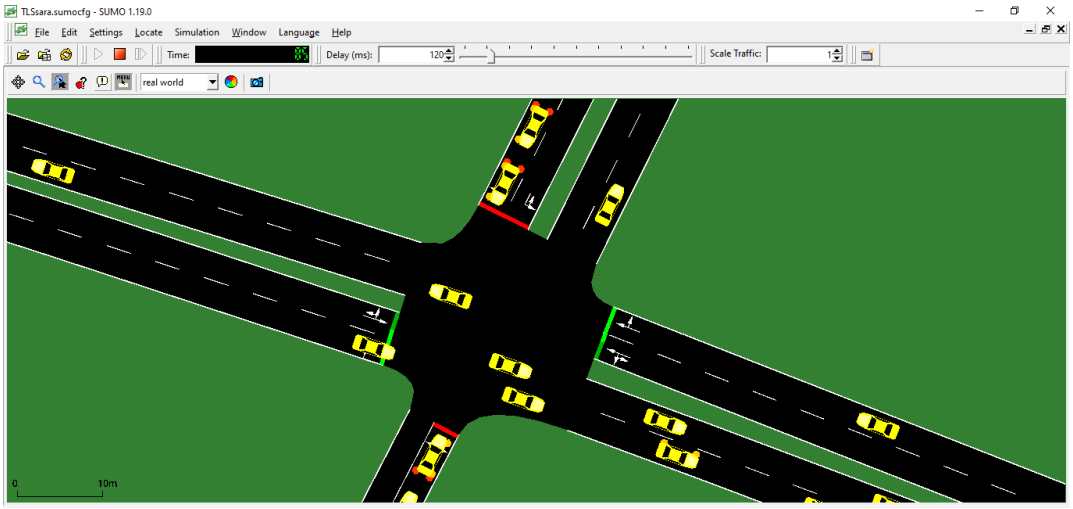

FIGURE 4.6 – Simulation en cours d'exécution

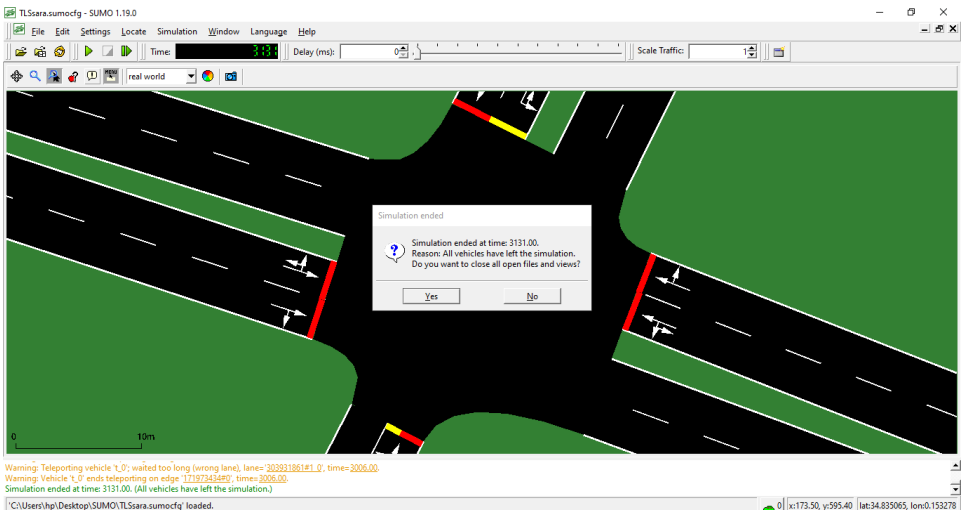

FIGURE 4.7 – Fin de la simulation

Une fois la simulation terminée, un fichier vehicle_flow_data sera créé dans le même répertoire.

	A	B	C	D	E	F	G	H	I
1	Time	Phase	Direction	Vehicle Flow	Phase Flow	Lane Flow Rate	Phase lane flow	Intersection Saturation	
2	1	0	Nord	0	0	0	0	0	
3	1	0	Nord	0	0	0	0	0	
4	1	0	Nord	0	0	0	0	0	
5	1	0	Nord	0	0	0	0	0	
6	1	0	Nord	0	0	0	0	0	
7	1	0	Nord	0	0	0	0	0	
8	1	0	Nord	0	0	0	0	0	
9	1	0	Nord	0	0	0	0	0	
10	1	0	Nord	0	0	0	0	0	
11	1	0	Nord	0	0	0	0	0	
12	1	0	Nord	0	0	0	0	0	
13	1	0	Nord	0	0	0	0	0	
14	1	0	Nord	0	0	0	0	0	
15	1	0	Nord	0	0	0	0	0	
16	1	0	Nord	0	0	0	0	0	
17	1	0	Nord	0	0	0	0	0	
18	1	0	Nord	0	0	0	0	0	
19	1	0	Nord	0	0	0	0	0	
20	1	0	Nord	0	0	0	0	0	
21	2	0	Nord	1.62950882	1.62950882	0.81475441	0.81475441	0.000452641	
22	2	0	Nord	1.62950882	3.25901764	0.81475441	1.629508819	0.000905283	
23	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
24	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
25	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
26	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
27	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
28	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
29	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
30	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	
31	2	0	Nord	0	3.25901764	0	1.629508819	0.000905283	

vehicle_flow_data

FIGURE 4.8 – Le fichier vehicle_flow_data

4.4 Présentation de l'application

Le but de notre application est d'ajuster la synchronisation des feux de circulation en fonction des conditions de circulation. Cela permet de réduire les embouteillages, d'améliorer les temps de parcours et de réduire le temps de retard et les émissions.

Dans cette section, nous décrivons les différentes fenêtres constituant l'application.

4.4.1 Fenêtre d'accueil

Lorsque l'utilisateur lance l'application, la première qui s'affiche est la fenêtre d'accueil (figure 4.8). Cette fenêtre lui donne la possibilité d'afficher la fenêtre suivante sur le bouton Démarrer.

FIGURE 4.9 – La fenêtre d'accueil

4.4.2 Fenêtre de paramètres :

Cette fenêtre qui représenté dans la figure 4.10 intitulée "Paramètres", est une fenêtre conçue pour afficher les paramètres associés a fonction de fitness (q,N et D) avec deux boutons représentant les directions cardinales (Nord/Sud, Est/Ouest) permettent d'afficher les valeurs spécifiques à chaque direction. Un autre bouton "Optimiser" ouvre une nouvelle fenêtre appelée "Résultat".

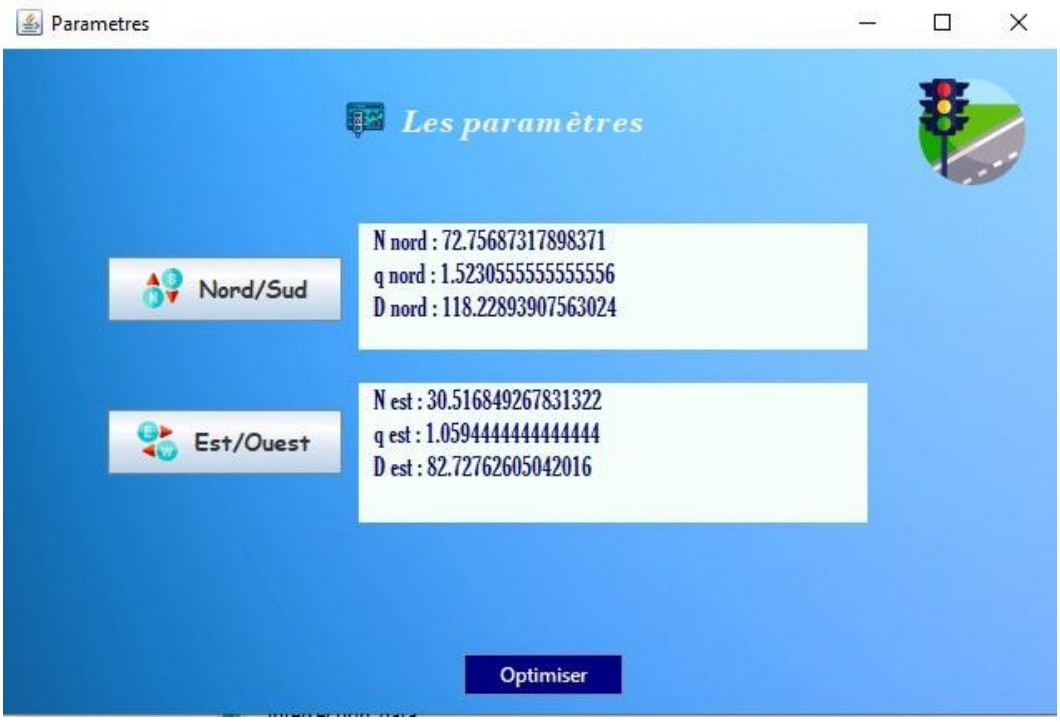

FIGURE 4.10 – La fenêtre des paramètres

4.4.3 Fenêtre de résultat :

L'interface Résultat affiche le meilleur chromosome (temps de vert) et le meilleur fitness obtenu par l'algorithme d'optimisation.

FIGURE 4.11 – La fenêtre du résultat

Le meilleur chromosome représente la configuration optimale de la temporisation des feux tricolores identifiée par l'algorithme.

Il se compose des durées de feu vert pour chaque direction de circulation (Nord, Sud, Est et Ouest). Ces durées de feu vert sont déterminées par l'algorithme afin de minimiser la congestion du trafic et d'améliorer la fluidité de la circulation à l'intersection.

Le meilleur ajustement (fitness), quant à lui, est une valeur numérique qui reflète la performance du meilleur chromosome.

4.5 Résultats expérimentaux

Les résultats obtenus ont été présentés sous forme de graphe valorisant la performance de l'approche proposée (SO-GA) pour les différents cycles et sont exposés ci-dessous.

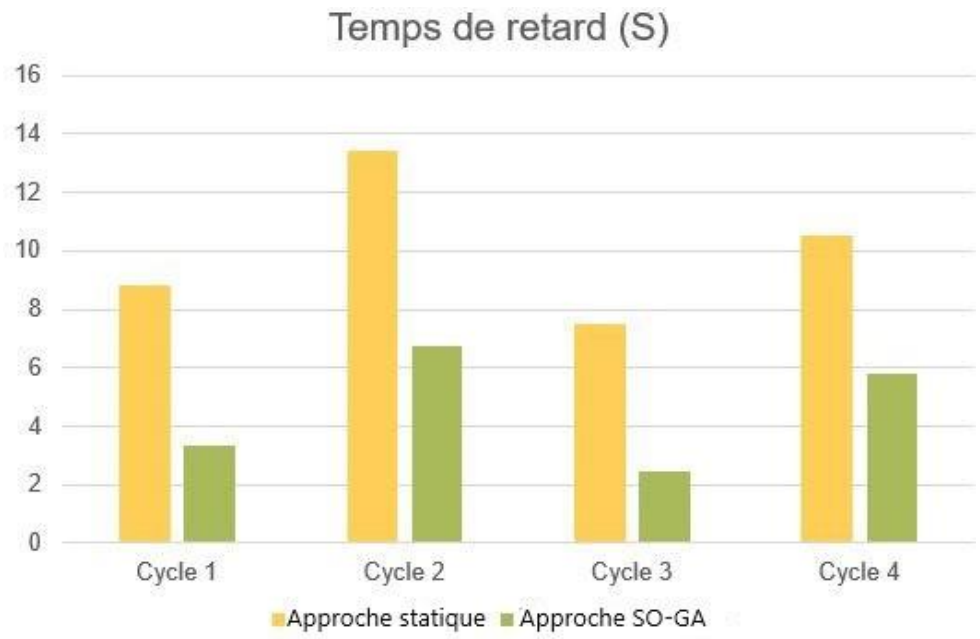

FIGURE 4.12 – Résultat de temps de retard

Le graphique présentant les temps de retard moyens pour différents cycles, basée sur des données de trafic en temps réel.

Le temps de retard "Fitness dynamique" (en vert) est généralement inférieur au temps de retard "Fitness statique" (en jaune) pour chaque cycle, ce qui indique que l'ajustement dynamique des temps de feux en fonction du trafic réel permet de réduire les retards globaux.

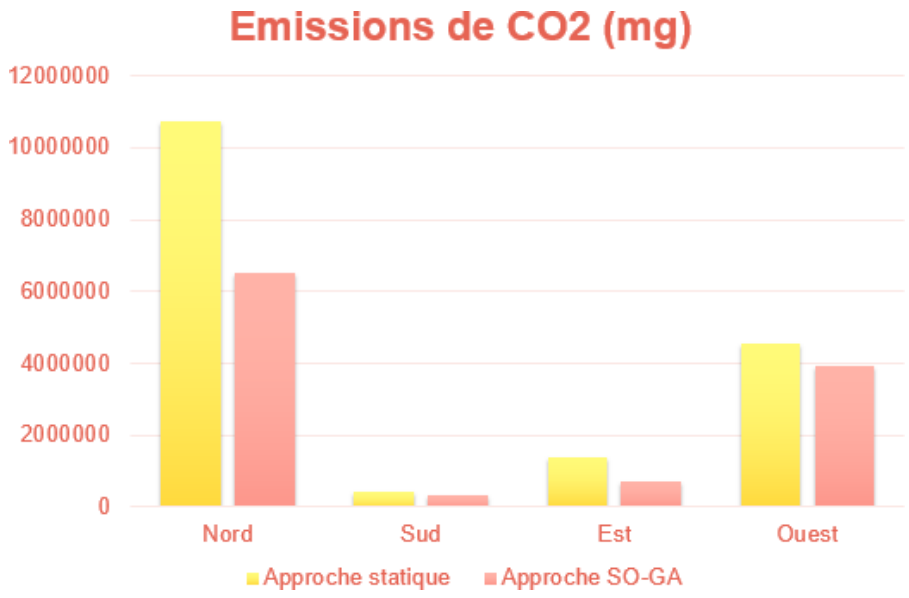

FIGURE 4.13 – Comparaison des émissions de CO_2

Le graphique compare les émissions de CO_2 statiques (en jaune) et dynamiques (en rose) dans les quatre directions (Nord, Sud, Est, Ouest).

Il montre que la stratégie dynamique réduit les émissions dans toutes les directions.

Les pourcentages de réduction des émissions de notre approche par rapport aux émissions de l'approche statiques pour chaque direction, affichés dans le tableau au-dessus.

Direction	Émissions d'approche statique	Émissions d'approche SO-GA	Réduction (%)
Nord	10727940	6510552	39%
Sud	431861.1	343710.7	20%
Est	1365324	731495.6	46%
Ouest	4559440	3907279	14%

TABLE 4.1 – Comparaison des réductions d'émissions

4.6 Discussion des résultats

Les données illustrées dans le graphique de figure 4.12 soulignent de manière significative les bénéfices de notre méthode d'optimisation des intervalles de signalisation lumineuse, en comparaison avec une méthode statique reposant sur des intervalles fixes et préétablis.

L'analyse de quatre cycles distincts révèle que le délai moyen, évalué à l'aide d'une fonction de fitness dynamique, est constamment plus bas que celui mesuré avec une fonction de fitness statique.

Cette différence marque clairement les avantages potentiels en matière de diminution des temps d'attente, grâce à l'ajustement des intervalles de signalisation en adéquation avec les conditions actuelles de circulation.

Ces observations confirment avec force la supériorité de notre stratégie hybride, qui intègre des algorithmes génétiques et l'algorithme d'optimisation snake pour une optimisation dynamique et en temps réel des durées des feux de signalisation. Cette stratégie offre une capacité remarquable d'adaptation dynamique, capable de réagir instantanément aux variations du trafic et de converger vers les configurations optimales de signalisation qui minimisent les temps de retard. Cela explique pourquoi les performances de notre approche surpassent largement celles d'une méthode statique.

Le graphique présenté dans la figure 4.13 permet de visualiser clairement les différences d'émissions de CO_2 entre les stratégies statique et SO-GA pour chaque direction.

Les réductions significatives des émissions dans les directions Nord et Est, particulièrement avec notre approche, sont mises en évidence. Cette réduction des émissions est probablement due à une gestion optimisée du trafic, qui favorise une circulation plus fluide et diminue les temps d'arrêt et d'attente, grâce à l'approche dynamique mise en œuvre.

Les résultats obtenus attestent que le modèle que nous proposons s'ajuste de manière efficace aux divers scénarios de trafic, ce qui se traduit par :

- Une calibration optimale des signaux lumineux, adaptée aux conditions de circulation.
- Une atténuation notable des embouteillages et de la congestion routière.
- Une amélioration conséquente de la fluidité du trafic.
- Une réduction appréciable des émissions de polluants atmosphériques.

4.7 Conclusion

Dans ce chapitre, nous avons décrit les outils techniques utilisés, le cas d'étude spécifique et l'application d'optimisation développée pour réguler le trafic à l'intersection étudiée.

En dernier lieu, nous avons analysé les résultats obtenus et avons conclu que l'approche utilisée, donne les meilleurs résultats de chaque exécution.

Conclusion Général

Les méthodes modernes de gestion du trafic aux intersections, notamment les feux de circulation, améliorent considérablement les conditions de circulation. Notre objectif était de concevoir et développer une nouvelle approche de gestion dynamique des feux de circulation basée sur des algorithmes meta-heuristiques.

En s'inspirant des principes de l'évolution naturelle et de la génétique avec l'algorithme génétique, ainsi que du comportement des snakes avec l'algorithme Snake Optimization (SO), cette approche vise à optimiser les temps de feux et les séquences de signalisation afin de fluidifier le trafic routier.

Les résultats préliminaires obtenus sont prometteurs et démontrent le potentiel de cette méthode pour réduire significativement les temps d'attente aux intersections, diminuer la congestion et les temps de trajet pour les usagers. Cela permettrait d'améliorer la qualité de vie en milieu urbain en réduisant le stress, le bruit et les risques d'accidents liés aux embouteillages.

En somme, cette étude ouvre la voie à de nouvelles pistes prometteuses pour relever le défi de la gestion optimale du trafic routier dans les villes modernes, en tirant parti des avancées dans le domaine de l'optimisation.

Dans les travaux futurs, il serait intéressant d'étendre notre approche à un cadre d'optimisation multi-objectifs afin de prendre en compte simultanément plusieurs critères contradictoires.

L'application de notre algorithme sur des réseaux routiers réels et l'intégration de données de trafic historiques permettraient également d'évaluer plus précisément ses performances dans des conditions opérationnelles.

Bibliographie

[1] Caron-Telders, C. (2020). La mobilité urbaine, le défi de la ville de demain. Les Echos.

[2] OCDE. (2010). Gérer la congestion urbaine. Organisation de Coopération et de Développement Économiques.

[3] https ://www.djazairess.com/fr/letemps/47827 (consulté le 03/04/2024)

[4] Mohamed Tlig, Coordination locale et optimisation distribuée du trafic de véhicules autonomes dans un réseau routier, Ecole doctorale IAEM Lorraine, mars 2015.

[5] Aidée Carriére, La gestion en temps réel d'un feu de circulation dans le contexte des systémes de transport intelligents , université de Montréal, Novembre 2005, Mémoire de maîtrise des sciences.

[6] https ://www.scienceabc.com/innovation/ready-steady-go-the-evolution-of-traffic- lights.html (consulté le 08/04/2024)

[7] https ://www.idrivesafely.com/defensive-driving/trending/history-and-meaning- colored-traffic-lights

[8] https ://www.theguardian.com/notesandqueries/query/0,5753,-1460,00.html

[9] Instruction interministérielle sur la signalisation routiére, Récupérer de http ://www.equipementsdelaroute.equipement.gouv.fr/IMG/pdf/IISR 6ePARTIE vc20120402 cle573dda.pdf.

[10] D.I. Robertson et R.D. Bretherton. "Optimizing networks of traffic signals in real time-the SCOOT method ". Dans : IEEE Transactions on Vehicular Technology 40.1, fév 1991.

[11] Association mondiale de la route / World Road Association. AIPCR Manuel sur les systémes de transport intelligents (STI) (seconde édition). Anglaise : ISBN 2-84060- 174-5 Route 2 Market Ltd. Fran¸caise : ISBN 2-84060-188-5 AIPCR Secrétariat. Paris, 2003.

[12] Vipin Kumar. “Algorithms for constraint-satisfaction problems : A survey ”. In : AI magazine 13.1 (1992), page 34.

[13] C. Lee. Fuzzy logic in control systems : fuzzy logic controller. ii. IEEE Transactions on Systems, Man and Cybernetics, 20(2) :419-435, mar/apr 1990.

[14] Z. R. Abdy. Fuzzy logic traffic signal control.

[15] F. Zou, B. Yang, and Y. Cao. Traffic light control for a single intersection based on wireless sensor network. In 9th International Conference on Electronic Measurement & Instruments (ICEMI 2009), pages 1-1040, 2009.

[16] M. Lescieux. Application à la commande floue. http ://auto.polytech.univtours.fr/automatique/AUA/ressources/Introduction logique floue.ppt.

[17] Techno-Science.net. (n.d.). Métaheuristique - Définition et Explications. Retrieved February 25, 2023, from https ://www.techno-science.net/glossaire-definition/Metaheuristique-fr-2022.htm

[18] Hertz, A. (2003). Les méta-heuristiques : quelques conseils pour en faire bon usage. In Optimisation de la production et gestion des flux (p. 1-15). Hermès Science Publications.

[19] Métaheuristique. (2023, 10 février). Dans Wikipedia, l’encyclopédie libre. Consulté le 25 février 2023, à l’adresse https ://fr.wikipedia.org/wiki/Métaheuristique

[20] Hassani, B. G. (2018). Fonctions de plusieurs variables. Cours de mathématiques. https ://www.mcours.net/cours/pdf/hassbg/hassbgli902.pdf.

[21] KHALDOUNA, Zahia. (2014). Simulation et optimisation des feux de circulation pour le flux des véhicules dans les zones à forte circulation (Thèse de doctorat, Université d’Annaba). https ://tel.archives-ouvertes.fr/tel-01234567

[22] Dréo et al., 2003 Dréo, J., Pétrowski, A., Siarry, P., & Taillard, E. (2003). Métaheuristiques pour l’optimisation difficile. Eyrolles.

[23] Scribbr. (2024). Algorithme d’optimisation par colonie de fourmis (ACO). https ://www.scribbr.fr/algorithme-aco/

[24] Scribbr. (2024). Algorithme d’optimisation par essaim de particules (PSO). https ://www.scribbr.fr/algorithme-pso/

[25] Singh, L., Tripathi, S., & Arora, H. (2009). Time Optimization for Traffic Signal Control Using Genetic Algorithm. International Journal of Recent Trends in Engineering, 2(2), 4-6.

[26] Zhang, H., Yuan, H., Chen, Y., Yu, W., Wang, C., Wang, J., & Gao, Y. (2021). Traffic Light Optimization Based on Modified Webster Function. Journal of Advanced Transportation, 1-10.

[27] Smith, J., & Johnson, A. (2020). "Application of Ant Colony Algorithm for Traffic Signal Timing Optimization." *Journal of Traffic Engineering*, 15(3), 45-58.

[28] Audet, O., & O'Carroll, P. (2019). Prédire la circulation à l'aide de l'apprentissage machine. Communication présentée lors du Congrès-Exposition conjoint ATC-STI Canada 2019, Halifax, Nouvelle-Écosse

[29] Cheng, R., Qiao, Z., Li, J., & Huang, J. (2023). Traffic Signal Timing Optimization Model Based on Video Surveillance Data and Snake Optimization Algorithm. Sensors, 23(15), 5157.

[30] Vogel, A., Goerick, C., & von Seelen, W. (2000). Evolutionary Algorithms for Optimizing Traffic Signal Operation. Honda Research Institute Europe GmbH. Retrieved from https ://www.honda-ri.de/

[31] Araghi, S., Khosravi, A., Creighton, D., Nahavandi, S. (2016). Influence of Meta-heuristic Optimization on the Performance of Adaptive Interval Type2-fuzzy Traffic Signal Controllers. Expert Systems With Applications, doi : 10.1016/j.eswa.2016.10.066.

[32] Qadri, S., Sultan, S., & et al. (2020). Optimisation des feux de signalisation pour la gestion du trafic urbain : une revue de la littérature. European Transport Research Review, 12(55). https ://doi.org/10.1186/s12544-020-00439-1

[33] Han, C., & Zhang, Q. (2008). Real-Time Detection of Vehicles for Advanced Traffic Signal Control. Dans 2008 International Conference on Computer and Electrical Engineering (pp. 590-594). IEEE1

[34] amirez-Polo, L., Jimenez-Barros, M. A., Varela Narváez, V., & Parodi Daza, C. (2022). Simulation and Optimization of Traffic Lights For Vehicles Flow in High Traffic Areas. Procedia Computer Science, 198, 548-553

[35] Hashim, F.A. ; Hussien, A.G. Snake Optimizer : A novel meta-heuristic optimization algorithm. Knowl.-Based Syst. 2022, 242, 108320.

[36] Yildizdan, G. (2023). Chaotic Snake Optimizer. AKÜ FEMUBUD, 23(055101), 1122-1141. https ://doi.org/10.35414/akufemubid.1263731

[37] Holland, J. H. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press.

[38] Z.MICHALEWICZ , K.DEB, M. SHMIDT , et TH. STIDSEN .Evolutionary algorithms for engineering applications, pp.73-94. Dans “Evolutionary algorithms in engineering and computer science : recent advances in genetic algorithms, evolution strategies, evolutionary programming, genetic programming and industrial a pplications ”, Edité par : Makela M., MiettinenK.,Neittaanmaki P. et Periaux J., Chichester, New York : Wiley, 500p, .1999.

[39] A.SOUQUET et F.G.RADET (2004). Algorithmes génétiques, TE de fin d’année, 50 p. Université de Nice Sophia Antipolis, Disponible sur : http ://souqueta.free.fr/Project/files/TE_AG.pdf .

[40] Mr. Nassir HARRAG.21/09/2011. "Optimisation des paramètres d’un réseau Ad Hoc par algorithmes génétiques". Ministere de l’Enseignement Superieur et de la Recherche Scientifique.Universite Ferhat ABBAS 3 Setif.

[41] Renders, J. M. (1995). Algorithmes Génétiques et Réseaux de Neurones. Paris, Hermès.

[42] Python Software Foundation. (2021). Python (Version 3.9) [Logiciel]. Disponible sur https ://www.python.org/

[43] Haggar, P. (2000). Practical Java : programming language guide. Addison-Wesley Professionnel.

[44] Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., & Wießner, E. (2018). Microscopic Traffic Simulation using SUMO. IEEE Intelligent Transportation Systems Conference (ITSC).