الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Nº Réf :……

Centre Universitaire

Abd Elhafid Boussouf Mila

Institut des Sciences et Technologie Département de Mathématiques et Informatique

Mémoire préparé en vue de l'obtention du diplôme de Master

En : Informatique

Spécialité : Sciences et Technologies de l'Information et de la Communication (STIC)

Technique de simulation-optimisation pour le contrôle des feux de signalisation via algorithme génétique

Préparé par : **BOUABDELLAH Samah**
AIOUADJ Amira

Soutenue devant le jury :

Encadré par	**TALAI Meriem**	**MAA**
Président	**HADJI Atmane**	**MAA**
Examinateur	**KIMOUCHE Abdelkader**	**MAA**

Année Universitaire : 2021/2022

Remerciements

Tout d’abord nous rendons grâce à Dieu, lui qui nous a permis d’être bien portant afin d’effectuer ce travail du début jusqu'à la fin.

Nous remercions nos parents respectifs pour leurs soutiens durant notre parcours d’études. Nos remerciements vont, à notre directeur de mémoire, le professeur Mme Talai Meriem, elle qui nous a guidés avec ses orientations, ses conseils et ses critiques tout au long de ce travail de recherche en nous laissant la liberté dont on avait besoins. On ne peut que lui être reconnaissant surtout pour ses qualités intellectuelles et humaines.

Nos remerciements vont aussi au membre du jury, d'avoir accepter d’évaluer ce travail et de participer à la soutenance.

Nous remercions Anwar qui nous a aidée beaucoup à la réalisation de ce mémoire

Enfin, nous sommes reconnaissantes envers tous les enseignants de l’Université Abd elhafidh Boussouf pour leur contribution à notre formation, et également à nos camarades, amis pour leurs aides précieuses.

Dédicace

A ma mère Fouzia, 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 cœur.

A mon père Kamel , qui a toujours cru en moi.

A ma chère sœur Ghada , mes chers frères Moncef et Ayoub

A mes chères amies Nesrin ,Rayan ,Rania,Dounia,Samah

A ma collègue Amira

SAMAH.

Dédicace

C'est avec un grand plaisir que j'apporte ce modeste Travail à tous ceux qui m'a gratifié de leur soutien et de leur Confiance.

Louanges à Dieu, qui m'a donné vie et santé pour le parachèvement de ce travail.

Et je voudrais exprimer toute notre gratitude à :

Ma chère maman **RAZIKA** qui m'a soutenue et encouragée durant les années d'études. Qu'elle trouve ici le témoignage de ma profonde reconnaissance.

A mon cher père **RACHID**, qui a toujours été là pour moi.

Mes chères sœurs: **SONIA, DOUNIA, MALAK**.

Mon cher frère : **OMAR**.

Mes petites : **SERINE, HOUDA**.

A l'homme, mon précieux offre du Dieu. Mon fiancé **AYMEN**

À mes belles amies : **LOUPI**, **HALIMA, SOUMIA, SOUHIR**, **AIDA, AYA**.

À tous mes collègues du Centre Universitaire de Mila, Mon amie,

Et binôme **SAMAH.**

À tous les membres de la famille **AIOUADJ** et **BOUREZG**.

AMIRA.

Résume

Cette recherche menée sur les carrefours du centre-ville de Mila, vise à fluidifier la circulation en réduisant le temps d'attente des véhicules et ainsi amortir les impacts économiques et environnementaux générés par la congestion routière. A cet effet, un algorithme génétique a été appliqué pour synchroniser le fonctionnement des panneaux lumineux et optimiser les durées de phase des feux de signalisation.

Abstract

This research carried out on intersections in downtown Mila, aims to fluidity traffic by reducing the waiting time of vehicles and in turn, cushion the economic and environmental impacts generated by traffic congestion. To this end, a genetic algorithm has been applied to synchronize the operation of light panels and optimize the phase durations of traffic lights.

الملخص

يهدف هذا البحث، الذي تم إجراؤه عند عدة تقاطعات في وسط مدينة ميلة ، إلى تبسيط حركة مرور السيارات من خلال تقليل وقت انتظار المركبات وبالتالي تخفيف الآثار الاقتصادية والبيئية الناتجة عن الازدحام المروري. ولهذه الغاية، تم تطبيق خوارزمية وراثية ستكون قادرة على مزامنة تشغيل الألواح الضوئية وتحسين أوقات مراحل إشارات المرور.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Le phénomène des embouteillages sur les réseaux routiers en milieu urbain est apparu depuis la deuxième moitié du vingtième siècle, notamment avec l'essor fulgurant des échanges économiques et sociaux. Ces échanges ont motivé l'augmentation du nombre des véhicules et boosté la demande sur les moyens de transport. Cela a eu des incidences sur l'organisation des activités et généré des pertes de temps, comme il a eu des retombées néfastes sur l'environnement ; car les embouteillages provoquent des émissions de gaz carbonique dans l'atmosphère de plus en plus importantes [1]. En 2002 à titre d'illustration, les usagers des routes urbaines dans la ville de Copenhague, la capitale du Danemark, ont passé 100 000 heures dans des embouteillages et perdu près de 750 millions d'Euros à cause des problèmes de circulation routière.

Aussi, les chercheurs essayent, depuis de longues années, à apporter des solutions aux problèmes des embouteillages à travers l'amélioration de l'état et du gabarit des routes et la construction de ponts et viaducs. Aussi, les pays développés consacrent annuellement d'importants budgets au secteur des transports en vue de préserver les réseaux routiers existants et y opérer des élargissements dans le but de fluidifier le trafic routier. Toutefois, le problème est loin d'être résolu, car on est souvent confronté au manque d'espace en milieu urbain et à l'insuffisance des ressources financières.

A la lumière de toutes ces difficultés, des approches basées sur le recours à la science ont vu le jour. Ces approches tentent d'améliorer la circulation routière en milieu urbain à travers la maitrise du fonctionnement des feux tricolores et la régulation de leur temps de manière plus performante. Mais malgré tous les efforts, la maitrise des feux demeure un problème complexe, compte tenu du caractère souvent anarchique du trafic routier.

Les chercheurs et les ingénieurs ont développé des instruments de simulation de la circulation routière pour comptabiliser les valeurs de la vitesse, de l'accélération, du ralentissement et de la position des véhicules. Et pour la maitrise de la circulation routière, l'emploi de systèmes informatiques puissants est de plus en plus envisagé. Aussi, l'amélioration réflective de la circulation par l'emploi de l'algorithme génétique s'avère, désormais, comme la clé au phénomène des embouteillages routiers.

Dans cette recherche, on a constaté l'état des routes au centre-ville de Mila et on a procédé à sa représentation au simulateur SUMO. On a étudié trois panneaux de régulation de la circulation. Chaque panneau comporte un feu rouge et un feu vert, fonctionnant sur des plages horaires allant de 10 à 30 secondes par phase. L'objectif de l'étude est de régler et améliorer le temps d'attente des véhicules et maitriser le flux des véhicules par l'emploi de logarithmes génétiques.

On a donc étudié le comportement de ces signaux lumineux sur le terrain et on a comparé ce comportement aux résultats obtenus par l'étude. On a constaté qu'il y a bel et bien une grande amélioration du temps de fonctionnement des signaux lumineux et une nette réduction du temps d'attente des véhicules. Et c'est là une confirmation incontestable de l'efficience de cet algorithme dans l'amélioration du rendement des feux de signalisation routière.

Le présent travail est organisé en 4 chapitres ;

Le premier chapitre s'intéresse à la technique d'optimisation-simulation principalement utilisée pour résoudre les problèmes du monde réel caractérisé par la complexité d'une modélisation mathématique classique.

Le second chapitre est une présentation succincte des principes de l'algorithme génétique.

Quant au troisième chapitre, il s'intéresse aux outils de simulation du trafic routier.

Ces trois chapitres constituent une introduction théorique des connaissances nécessaires pour aborder le problème de perte de temps sur la circulation urbaine.

Le quatrième chapitre présente une synthèse des réflexions et de l'implémentation de la solution proposée au problème cité.

Finalement, le cinquième chapitre illustre les résultats obtenus via des schémas et diagrammes divers.

CHAPITRE 1 : Les techniques d'optimisation-simulation

1. Introduction :

L'objectif principal de ce chapitre est de présenter une technique d'optimisation simulation. D'une manière générale l'optimisation peut contribuer à résoudre certains problèmes dans différents domaines comme celui de la régulation du trafic routier via des feux de signalisation. Donc la simulation est considérée comme une étape nécessaire pour les gens qui travaillent sur les projets d'aménagement routier ou sur la régulation des flux, en utilisant le simulateur Sumo.

2. Optimisation :

2.1 Définition :

L'optimisation est à l'origine une branche des mathématiques, cherchant à analyser et à résoudre analytiquement ou numériquement les problèmes qui consistent à déterminer le meilleur élément d'un ensemble, au sens d'un critère quantitatif donné. Ce mot vient du latin optimum qui signifie le meilleur. L'optimisation est le processus de trouver les conditions pour satisfaire le maximum ou le minimum d'une fonction[2].

L'optimisation joue un rôle important en ingénierie quel que soit le domaine. Aujourd'hui, tous les systèmes susceptibles d'être décrits par un modèle mathématique sont optimisés. La qualité des résultats et des prédictions dépend de la pertinence du modèle, de l'efficacité de l'algorithme et des moyens pour le traitement numérique. L'optimisation est souvent attribuée aux ingénieurs[2].

L'optimisation des systèmes en ingénierie (génie civil, génie mécanique et énergétique, génie chimique, génie informatique et télécommunication, génie électrique, etc.) permet de trouver une configuration idéale, d'obtenir un gain d'effort, de temps, d'argent, d'énergie, de matière première, ou encore de satisfaction[2].

Lorsqu'un système est difficile à optimiser à cause de sa complexité, il est convenable d'en optimiser les sous-systèmes et de choisir la combinaison optimale de ces sous-systèmes[2].

2 .1 Représentation mathématique d'un problème d'optimisation :

2.2.1 Fonction objectif :

La résolution d'un problème d'optimisation suppose en préalable, la connaissance formelle d'une fonction objective, appelée parfois fonction coût exprimée ici par une fonction y telle que :

$$y= y(x_1, x_2, x_{3,\ldots,} x_n) \qquad (1.1)$$

Cette fonction dépend d'un vecteur $\{x_i\}$, faisant intervenir n variables. L'optimum recherché est soit un minimum noté Min[y], soit un maximum noté Max[y]. Mais la résolution du problème se résume en fait à la recherche d'un extremum, car il y'a formellement identité entre les deux problèmes. En effet, on montre facilement que :

$$\text{Max}\,[y(x_1, x_2, x_{3,\ldots,} x_n)] = \text{Min}\,[-y(x_1, x_2, x_{3,\ldots,} x_n)] \qquad (1.2)$$

Par ailleurs, l'adjonction d'une constante à la fonction objective n'affecte pas le point correspondant à l'optimum. On distingue deux cas de figure possibles qui sont :

- Optimisation sans contraintes.
- Optimisation avec contraintes.

2.2.2 Optimisation sans contraintes :

Les n variables précédentes sont définis parmi un ensemble de valeurs acceptables, susceptibles d'aboutir à l'optimum et sont indépendantes entre-elles. Alors l'optimisation entreprise est une optimisation sans contraintes. Ce problème est plus simple que le suivant.

2.2.3 Optimisation avec contraintes :

Dans ce cas, les variables mises en œuvre sont définies parmi un ensemble de valeurs acceptables, susceptibles d'aboutir à l'optimum mais qui ne sont pas indépendantes entre-elles. Les liaisons qui existent entre celles-ci peuvent se traduire par deux types de contraintes qui sont :

- Contraintes d'égalité
- Contraintes d'inégalité

✓ Contraintes d'égalités

Ces contraintes sont mises sous la forme d'un ensemble de m fonctions égales chacune à zéro :

$$\{\phi_i\} = \{\phi_i\,(x_1, x_2, x_{3,\ldots,} x_n)\} = \{0\} \qquad (1.3)$$

Pour que le problème posé puisse admettre une solution, il faut nécessairement que le nombre de contraintes m soit inférieur au nombre de variables n. Dans le cas limite où m = n, le système est indéterminé ; son point d'état étant fixe, la recherche d'un optimum n'a aucun sens. De même, si m>n, le problème est mal posé.

✓ Contraintes inégalités

Ces contraintes s'expriment sous la forme :

$$\{\psi_i\} = \{\psi_i\,(x_1, x_2, x_{3,\ldots,} x_n)\} \leqq \{k_i\} \qquad (1.4)$$

Où $\{k_i\}$ représente le vecteur des constantes du second membre de (1.4)

Dans la majorité des problèmes posés, ces contraintes apparaissent et traduisent des limitations physiques ou technologiques apparaissant dans le système. Citons par exemple :

- Des seuils de température supérieurs ou inférieurs,
- Des contraintes mécaniques maximales,

- Des vitesses de rotation maximales,
- Des limitations en puissance,
- Des seuils de débit,
- Des durées de vie, etc… [2]

2.3 Méthodes d'optimisation :

Dans ce qui suit, nous énumérons quelques grandes familles d'optimisation qui sont développées dans la littérature, ainsi que leur caractérisation succincte. Les méthodes d'optimisation sont classées selon la référence [2] comme suit :

2.3.1 Calcul des variations – multiplicateurs de Lagrange et méthodes dérivées (équations d'Euler, Hamilton, Jacobi) :

Ces méthodes fonctionnent avec des contraintes de type Egalite. Elles nécessitent le calcul des dérivées partielles de la fonction objectif et des contraintes.

2.3.2 Méthodes de recherche :

Ces méthodes parfois appelées méthodes d'exploration directe se satisfont de l 'examen en des points discrets de la fonction objectif. Elles sont adoptées lorsque les composantes sont connues de façon discrète.

Parmi ces méthodes, on peut citer :

- Les méthodes d'énumération,
- Les méthodes d'exploration au hasard,
- les méthodes dérivées de l'exploration à une seule variable, en particulier la méthode de Fibonacci,
- Les méthodes de simplex
- Les méthodes d'exploration par alternance de variables,
- Les méthodes de Hookes-Jeeves, de Powell, etc [2].

2.3.3 Programmation linéaire :

C'est la méthode la plus employée surtout pour les grands systèmes, mais cette méthode nécessite une fonction objective et des contraintes linéaires[2].

2.3.4 Programmation géométrique :

Probablement c'est l'une des méthodes les plus récentes. La fonction objectif associée doit être un polynôme ou les variables apparaissent avec des exposants entiers ou non (il est à remarquer que ce cas est courant dans les ajustements de fonction) [2].

2.3.5 Programmation dynamique :

Cette technique encore appelée **optimisation dynamique** a été développée par Bellman. Elle permet de déterminer une fonction optimale, plus qu'un point d'état optimum. On obtient alors la *trajectoire optimale* sous forme d'une fonction reliant plusieurs variables d'état[2].

Cette méthode n'est pas sans lien avec le calcul des variations ; la programmation dynamique procède toutefois par une série de processus discrets, alors que le calcul des variations donne un résultat continu.

2.4 Elaboration du modèle pour un problème d'optimisation :

La mise en forme d'un problème d'optimisation est sans doute la partie la plus délicate de l'optimisation. Dans ce qui suit, l'énumération des principales étapes suivies dans la résolution d'un problème.

2.4.1 Enumération des étapes :

On peut citer dans l'ordre chronologique des actions à mener pour l'optimisation d'un système ou d'un procède :

- Choix de la configuration du système,
- Explication des variables,
- Spécification des caractéristiques des composants,
- Écriture des équations relatives aux entrées et sorties,
- Choix du critère d'optimisation et de recherche de la fonction objectif,
- Contraintes diverses liées au modèle mis en œuvre, aux équations de liaisons entre composants, aux conditions mathématiques d'existence et toute autre contrainte souhaitée.

(Ces deux dernières phases sont souvent difficiles).

- Élimination de variables d'état et de contraintes d'égalité par la méthode de substitution lorsque cette élimination est possible (réduction des équations de contrainte)
- Choix de la méthode d'optimisation

Si la technique le nécessite, recherche d'un vecteur d'état initial relatif aux variables indépendantes. [2]

2.4.2 Analyse des résultats :

Un ingénieur ou un chercheur doit toujours examiner la solution obtenue pour le problème à la lumière de son expérience et de son sens pratique ou physique, pour voir si la solution obtenue est acceptable[2].

2.5 Informatique et optimisation :

L'optimisation n'est pas un domaine de recherche purement mathématique. En effet, l'informatique intervient aussi bien dans la création d'algorithme d'optimisation que dans la programmation de méthodes mathématiques pour l'optimisation. Aussi, quel que soit le mode d'utilisation des moyens informatiques (en mode différé ou en mode interactif), le concepteur du programme ou du logiciel doit soigneusement commenter son travail et même s'investir dans la rédaction d'un manuel d'utilisation[2].

3. Simulation des systèmes réels :

3.1 Définition :

La Simulation se définit comme un outil à la fois scientifique et informatique qui permet l'exploitation d'une représentation virtuelle - ou modèle - du comportement d'un objet, d'un système, d'un phénomène, d'un processus industriel ou d'une organisation. La mise en œuvre de ces modèles par la simulation a généralement pour but d'analyser et comprendre le fonctionnement de systèmes complexes, de façon à prévoir et contrôler leur comportement dans tout son cycle de

vie . Elle présente de nombreux atouts ; d'ordre économique tout d'abord car elle permet d'avoir une meilleure réactivité, une meilleure capacité d'anticipation et d'apporter des gains de compétitivité [6].

3.2 Intérêt de la simulation :

La simulation est d'un apport considérables résumé en les points suivants :

1) Bonne manière de rassembler systématiquement des données pertinentes. Cela contribue à une large connaissance des caractéristiques des systèmes et de leurs opérations.
2) Permet de voir les variables importantes et comment elles sont reliées. Cela peut mener éventuellement à des formulations analytiques pertinentes.
3) Parfois on souhaite connaitre les distributions de probabilité plutôt que seulement les moyennes et les variances.
4) Peut parfois permettre de vérifier une solution analytique incertaine.
5) La simulation coute moins cher que de faire des expériences.
6) La simulation donne un contrôle sur le temps. Il est possible d'étudier des effets sur des périodes de temps longues ou au contraire de passer au ralenti certains événements.
7) La simulation est sans danger. On peut étudier divers effets sans déranger les usagers.
8) Choisir entre différentes alternatives de conception, en testant de nouvelles ou raffiner les modélisations analytiques.
9) Être intégré dans d'autres outils (pour l'étude d'optimisation d'une circulation par exemple) 10) Entraîner le personnel hors risques du domaine simulé.

3.3 Utilisation des modèles de simulation :

L'appel aux modèles de simulation sur ordinateur des systèmes réels est à priori un choix de modélisation du système étudié. Ce choix repose sur les objectifs de la modélisation, des compromis entre coûts permis ou souhaités, des durées d'utilisation à long ou court termes et de la précision souhaitée. Mais il devient parfois nécessaire ou même inévitable, comme par exemple :

- Lorsque les modèles mathématiques sont inapplicables à cause des échelles temporelles et spatiales ou bien de la complexité de la situation.
- À cause des hypothèses sous-jacentes, on doute de la précision et de l'applicabilité des résultats donnés par les procédures mathématiques et heuristiques classiques.
- Quand les mathématiques donnent un résultat inacceptable (résultat statique alors qu'il devrait être dynamique).
- Lorsqu'il y a un besoin de visualiser la dynamique des systèmes.

3.4 Développement d'un modèle de simulation :

Les principales étapes de développement d'un modèle de simulation sont les suivantes : [3]

3.4.1 L'identification du problème :

Il s'agit de définir le problème à priori ; de déterminer quelles sont les sorties désirées, quelles entrées affectent ces sorties, les limites temporelles et spatiales du problème, de déterminer les éléments stochastiques intervenant dans la procédure à simuler, définir la dynamique du système dans le temps …etc[3].

3.4. 2 Etudier l'apport et la nécessité de la simulation :

Pour effectuer cette étape, Il faut se poser certaines questions telles que : comment le problème pourrait-il être résolu sans simulation ? Pourquoi la simulation est-elle la meilleure solution ? Existe-t-il d'autres méthodes pour étudier le système en question ? La simulation étant choisie peut-elle effectivement apporter satisfaire les objectifs souhaités ? [3].

3.4. 3 Formulation du problème :

Cette étape a pour objectif de montrer explicitement les activités du système en incluant les entrées et sorties déterminées précédemment et leur traitement. Il en résulte des graphiques composés des activités reliées. En fin de cette étape, on détermine toutes les entrées dont on a besoin pour les activités réalisées par le modèle et avec précision et on y associe les sorties, ce qui permet de collecter les données à l'étape suivante [3].

3.4. 4 Collecte et entrée des données :

La collecte des données se base sur les entrées et sorties définies précédemment. On procède comme suit :

- Observer le système réel et, au début collecter, seulement un échantillon de données.
- Etudier les données collectées pour qu'elles soient compatibles avec les besoins du modèle de simulation [3]
- Faire un plan de collecte des données.
- Adapter la taille des échantillons à la calibration et la validation.
- Il est toujours utile de faire une revue de littérature sur les simulations et études précédentes.

3.4. 5 Formulation du modèle mathématique :

Cette formulation est la plus importante et la plus longue car elle-même passe par trois étapes :

- On se concentre sur la procédure qui relie l'entrée à la sortie.
- On identifie ensuite les principales sous-procédures et leurs relations.
- On détaille chaque sous-routine.

Cette étape nécessite également l'estimation des paramètres utilisés dans les formulations mathématiques. Certains paramètres sont déterministes alors que d'autres sont stochastiques.

- ✓ **Paramètres déterministes** : peuvent être constants pour toutes les situations, prendre différentes valeurs constantes selon le type de situation ou peuvent varier de manière continue selon certaines formes de régression.

- ✓ **Paramètres stochastiques** : ont besoin d'un centre et d'une déviation ainsi que d'une forme de distribution identifiée. Peut être très simple ou au contraire très complexe. La balance entre réalisme et simplicité est souvent testée à ce niveau de la simulation. [3]

3.4. 6 Evaluation de l'état actuel du modèle :

Réaliser des solutions calculées à la main pour identifier les boucles incomplètes ou les boucles fermées, pour vérifier la flexibilité de l'entrée des données et des valeurs et pour s'assurer de la cohérence des résultats. On peut alors modifier des variables, des paramètres ou la structure du modèle.

A ce stade, on peut décider d'accepter ou de rejeter le modèle développé. Si on rejette le modèle, on peut toujours réutiliser certains éléments.

3.4. 7 Rédaction de code informatique :

Cette étape devient relativement simple si la cinquième étape est effectuée correctement. Il est important de choisir le bon langage informatique. Ce choix repose sur quelques facteurs tels que la connaissance des langages informatiques, la cohérence des propriétés du modèle avec le langage de simulation choisi et la façon dont d'autres personnes utilise le modèle de simulation par rapport au services des autres. Si le programme doit être utilisé et éventuellement modifié par d'autres personnes, il est important d'inclure de nombreux commentaires dans le programme. on passe évidemment par une étape de débogage, qui inclut quelques une ou toutes les activités suivantes : déboguez les sous-routines une à une et les assembler ensuite, utilisez des données déterministes initialement connues au lieu de données aléatoires ainsi qu'effectuer des calculs de vérification manuels. Ajoutez temporairement des intermédiaires.

Le code ainsi obtenue approche la réalité à travers les fonctions de base souhaitées par la modélisation. Il ne produit exactement toutes les fonctions objectives de la modélisation.[3]

3.4. 8 La validation :

Cette étape comprend trois sous-étapes, il s'agit de :[3]

- ✓ **La vérification** est une phase complémentaire du débogage précédent du code. L'objectif est de s'assurer que le code réalise exactement ce que l'on souhaite dans la modélisation.
- ✓ **La calibration** dépend de la qualité des relevés sur le terrain, de la facilité de compréhension du modèle et de la complexité du modèle étudié. Seule une partie des données peut être utilisée pour la calibration. On peut réaliser des ajustements pour que les sorties correspondent aux observations.
- ✓ **La validation** permet de comparer les données calculées par l'ordinateur et les données mesurées sur le terrain. Aucun ajustement n'est fait au modèle et les différences montrent dans quelle mesure le modèle représente la réalité avec les conditions choisies.

Si les résultats de la validation sont acceptables, le modèle de simulation est prêt pour les applications. Si non, il faut effectuer d'autres calibrations et validations.

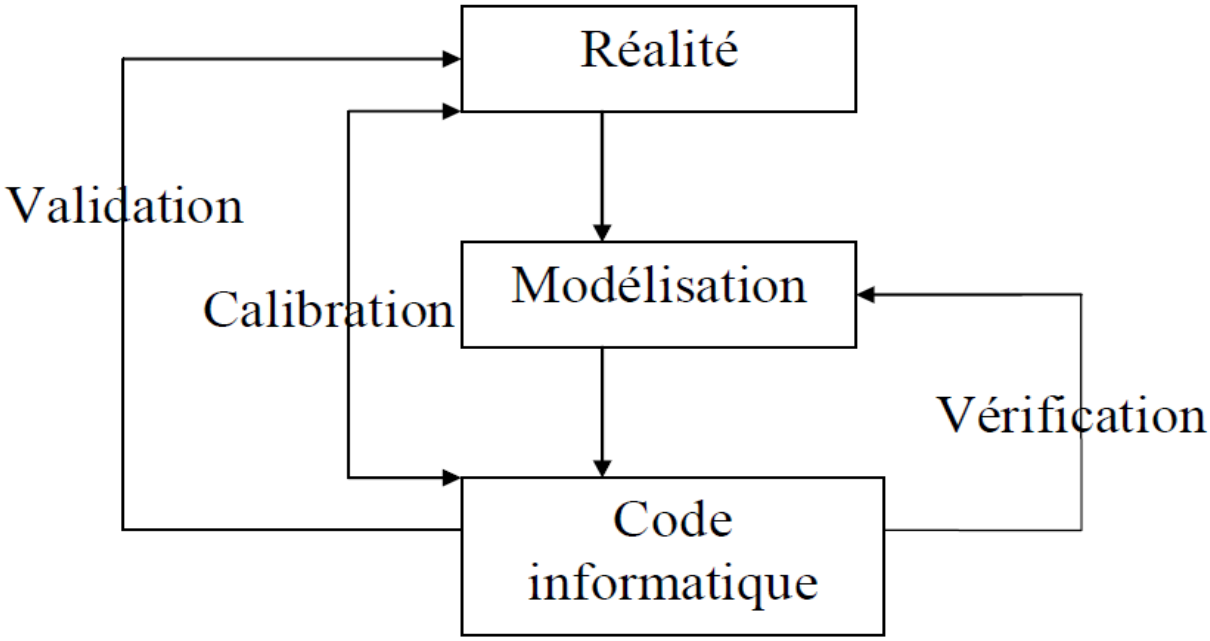

Figure 1 : La vérification, La calibration et La validation dans la modélisation.

3.4. 9 Conception des essais :

Toutes les applications issues de modèles de simulation nécessitent un cadre d'expérimentation. Sa compréhensibilité dépend de la taille et de la flexibilité du modèle ainsi que de la complexité et de la variété des situations à évaluer.

3.4. 10 Production et analyse des résultats, documentation :

Toutes les simulations devraient être sauvegardées et le code devrait offrir la possibilité de nommer toutes les sorties de résultat.

L'analyse des résultats peut révéler que le code doit être modifié ou qu'il faut utiliser d'autres logiciels pour représenter graphiquement les résultats, pour réaliser des tests statistiques ou pour mettre en forme les résultats.

La documentation ne devrait pas seulement contenir les résultats obtenus. Elle devrait également contenir un manuel pour l'utilisateur et pour l'opérateur du système. Toutes les variables devraient être listées et définies, un diagramme de fonctionnement devrait être inséré, un exemple de problème contenant des entrées et des sorties devrait être ajouté. [3]

3.4. 11 Analyse des résultats de la simulation :

La plupart du temps, beaucoup de ressources sont allouées (de façon plus ou moins directe) au développement du modèle de simulation mais peu à l'analyse de ses sorties.

Attention lorsque le modèle est heuristique : plusieurs sorties sont nécessaires pour analyser correctement la situation.[3]

3.5 Modèles de simulation pour le trafic routier :

Nous présentons ici quelques classifications des modèles de simulation selon des critères spécifiques.

3.5.1 Première distinction : changement des états des scénarios :

Pratiquement tous les modèles de simulation de feux de signalisation sont dynamiques et ont pour variable indépendante de base le temps.

Il existe cependant deux principaux types de modèles :

- Continus (changement continu d'état avec stimuli continu)
- Discrets (succession d'états changeant brusquement d'un instant à son suivant)

Les modèles discrets peuvent pour leur part être :

- Discrets dans le temps (division du temps en intervalle de longueur constante). Les calculs sont effectués pour chacun des intervalles.
- Discrets dans les événements (division du temps en période s'écoulant entre deux événements connus – circulation par exemple). Ce type de modélisation permet de gagner du temps de calcul mais ne peut être avantageusement utilisé que lorsque la circulation change peu d'état ou que la taille du système étudié reste relativement petite. [3]

3.5.2 Deuxième distinction : niveau de perception du trafic

On peut également classer les modèles selon le niveau de détails qu'il propose. Ainsi, on retrouve :

- Modèles microscopiques (MI). Ils rendent compte, à un haut niveau de détails, des objets du système ainsi que de leurs interactions. (ex : changement de voie, selon l'interaction entre véhicules impliqués)
- Modèles mésoscopiques (ME). Ils représentent la plupart des objets à un haut niveau de détails mais leurs interactions à un niveau de détails relativement moindre. (ex : changement de voie, selon densité relative des voies)
- Modèles macroscopiques (MA). Représentent à un faible niveau de détails les objets et leurs interactions. Perçoivent le flux de trafic comme un flux liquide s'écoulant dans un canal tel par exemple le flux de circulation par voie où la représentation est agrégée et ne visualise donc pas la manœuvre de changement de voie. [3]

3.5.3 Troisième distinction : types de variables

Une autre classification permet de distinguer les modèles de simulation selon le type des variables utilisées :

- Modèles déterministes : Aucune variable aléatoire ; les relations sont de type mathématique, statistique ou logique.
- Modèles stochastiques : Incluant des fonctions de probabilités.

3.6 Le simulateur SUMO :

SUMO (Simulation of Urban Mobility) est un logiciel de simulations de trafic routier open source sous licence ONU public (OPL), dont le développement a commencé 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.

La Figure I-10 montre le progiciel SUMO qui contient une suite d'applications qui aident à préparer et à exécuter la simulation d'un scénario de trafic routier. Il est très portable et nécessite uniquement l'installation des bibliothèques C++. Dans SUMO, chaque véhicule a son propre chemin, le comportement du véhicule est vivant comme le changement de voie. Les routes dans SUMO sont présentées sous forme de plusieurs voies. Il y a des intersections à base de règles de circulation et d'autres à priorité [4]

3.7 Techniques d'optimisation-simulation :

3.7.1 Présentation :

De nombreux problèmes d'optimisation difficiles nécessitent l'évaluation d'une fonction objectif boîte noire, coûteuse en temps et en ressources de calcul. Pour faire face à cette difficulté, une technique classique consiste à apprendre un modèle de substitution, également appelé méta-modèle. La substitution revient à formuler des modèles mathématiques rapides à évaluer qui approximent la fonction objective, afin de limiter le recours à son évaluation. Ainsi, à chaque étape du processus d'optimisation, le méta-modèle permet la recherche rapide de solutions prometteuses. Cependant, le nombre considérable de travaux réalisés aussi bien dans le domaine de simulation que celui de de l'optimisation a permis l'émergence d'un domaine de recherche à part entière nommé : « Optimisation via Simulation ». L'objectif de l'OvS est de trouver la ou les solutions optimales en simulant le problème à optimiser. En ce sens, elle peut être considérée comme un lien entre les deux domaines de recherches que sont : « l'optimisation » et « la modélisation et la simulation ».

Une approche d'optimisation via simulation comprend généralement un module d'optimisation qui a pour rôle de guider la direction de recherche des solutions et un module de simulation utilisé pour l'évaluation des performances des solutions candidates suggérées par le module d'optimisation. Elle se concentre nécessairement sur la cohérence des échanges entre les deux modules.

Par rapport aux méthodes de programmation mathématiques, les méthodes d'optimisation basée sur la simulation remplacent la fonction d'objectif analytique par un ou plusieurs modèles de simulation. Les variables de décisions correspondent aux conditions sous lesquelles les modèles de simulation sont exécutés.

3.7. 2 Quelques travaux

Dans la littérature, de nombreux travaux de recherche ont été réalisés pour le développement et l'utilisation des méthodes d'optimisation basées sur la simulation. On cite à titre d'exemples les travaux suivants :

J. Kim et B. Zeigler [6] envisagent dès 1996 l'utilisation de la modélisation et de la simulation pour l'optimisation de problèmes complexes. A partir d'une conception nommée « Hierarchical Distruted Genetic Algorithms » (HDGA), l'espace de recherche est analysé à différentes échelles formant une hiérarchie de clusters eux-mêmes répartis dans différentes couches.
Les auteurs de [5] proposent en 2003 une étude qui se base sur la création d'un outil unique d'aide à la décision intégrant la modélisation et la simulation de modèles existants nommés DEVS. À travers un langage unique, l'auteur permet l'utilisation de différentes méthodes d'optimisation.
En 2011 deux auteurs [5] proposent un espace de travail nommé «SImulation-based Multi-objective Evolutionnary OptimizatioN » (SIMEON) permettant l'optimisation de problèmes multicritères représentés sous la forme de modèles DEVS.

3.8 Avantages et inconvénients :

Plusieurs études ont été menées pour l'analyse des avantages et des inconvénients de l'utilisation de la simulation par rapport à l'optimisation mathématique et dans différents domaines d'application. Les auteurs recommandent d'utiliser conjointement les techniques d'optimisation et de simulation pour bénéficier des avantages suivants :

Lors de l'optimisation dans un système donné, la simulation peut aider en évaluant l'impact ou les impacts d'utilisation des différentes politiques.

De même, de nombreux événements aléatoires en termes de la variation de leur temps d'occurrence et les fluctuations des entrées ont des influences majeures sur la performance souhaitée du système et doivent être considérés par le décideur pour une évaluation plus réaliste. Ce réalisme est assuré par des outils de simulation spécialisés et évolués.

Cependant des limites sont présentes concernant le choix de la méthode d'optimisation ainsi que la généricité de l'outil :
Les différentes approches utilisent un nombre très limité de techniques d'optimisation. Le choix n'est jamais justifié puisque effectué de manière empirique et ne permet pas la comparaison des performances. Le problème d'efficience de ces techniques se pose puisque la méthode d'optimisation est adaptée aux spécificités du problème étudié ce qui ne permet pas sa réutilisation pour d'autres applications ultérieures. Les phénomènes de dépendance entre les différents modules sont importants et nécessitent des temps d'adaptation considérables et de bonnes connaissances informatiques.
La simulation de certains systèmes nécessite des temps de calcul considérables. L'optimisation de certains problèmes complexes nécessite elle aussi des temps de calcul importants. Le regroupement de processus de simulation et d'optimisation peut alors être générateur de temps de calcul dissuasifs pour de nombreux décideurs.

4. Conclusion :

La technique d'optimisation-simulation permet de générer directement des solutions aux problèmes d'optimisation, il suffit d'élaborer des simulations réalistes et de choisir un algorithme adéquat au type de problème en question.

Le trafic urbain suscite, depuis des décennies, l'intérêt des recherches scientifiques. Différents simulateurs ont été créé et déployés. De même, les techniques d'optimisation ont largement été testées pour le contrôle du trafic routier urbain. Dans ce qui suit, nous nous intéressons davantage à ces outils de simulation ainsi qu'à l'algorithme choisie pour notre travail.

CHAPITRE 2 : Algorithme Génétique

1. Introduction :

Les algorithmes génétiques (AGs) appartiennent à la famille des algorithmes évolutionnaires. Leur but est d'obtenir une solution approchée à un problème d'optimisation, lorsque, pour le résoudre, il n'existe pas de méthode aboutissant à la solution exacte en un temps raisonnable. La solution est approchée par « *bonds* » successifs, comme dans une procédure de séparation et d'évaluation[7].

Ce n'est que vers la fin des années 60 que John Holland et son équipe, relayés plus tard par d'autres chercheurs comme Goldberg [8] et Davis [9] ont pu adapter ces algorithmes à la recherche des solutions pour des problèmes d'optimisation en faisant l'analogie entre un individu dans une population et une solution d'un problème dans un ensemble de solutions.

Les algorithmes génétiques appliqués à un problème d'optimisation font évoluer un ensemble de solutions candidates, appelé population d'individus. Un individu représente une solution possible du problème donné. A chaque individu est attribué une « *fitness* » qui mesure la qualité de la solution qu'il représente, souvent c'est la valeur de la fonction coût à optimiser. Ensuite, une nouvelle population de solutions possibles est produite en sélectionnant les parents parmi les meilleurs de la génération actuelle. La nouvelle population contient une plus grande proportion de caractéristiques des meilleurs individus de la génération précédente [7] .

Les algorithmes génétiques ont montré de nombreuses applications réussies à de nombreux domaines[7], notamment dans le contexte du contrôle du trafic routier. Nous nous intéressons dans ce chapitre aux caractéristiques de ces algorithmes car ils constituent la méthode de résolution adoptée dans notre travail.

2 . Applications des algorithmes génétiques :

Les algorithmes génétiques sont une abstraction sur la théorie d'évolution [8]. L'idée fondatrice est simple : si l'évolution a optimisé les processus biologique, l'utilisation du paradigme de l'évolution pour trouver des solutions optimales dans le cadre de traitement informatiques possède un sens.

L'utilisation de la théorie de l'évolution comme modèle informatique pour trouver une solution optimale peut se justifier par le fait que la théorie d'évolution permet la recherche de la solution parmi un très grand nombre de possibilités dans un laps de temps raisonnable [13] .

Les applications des AG sont multiples : optimisation de fonctions numériques difficiles (discontinues, multimodales, bruitées,…), traitement d'image (alignement de photos satellites,

reconnaissance de suspects,…), optimisation d'emplois du temps, optimisation de design, contrôle de systèmes industriels, apprentissage des réseaux de neurones, etc. Les AG peuvent être utilisées pour contrôler un système évoluant dans le temps (chaîne de production, centrale nucléaire…) car la population peut s'adapter à des conditions changeantes. En particulier, ils supportent bien l'existence de bruit dans la fonction à optimiser. Ils peuvent aussi servir à déterminer la configuration d'énergie minimale d'une molécule ou à modéliser le comportement animal. [13]

3. Fonctionnement des algorithmes génétiques :

3.1 Modélisation et résolution des problèmes réels via algorithme génétique :

Les algorithmes génétiques fournissent des solutions aux problèmes n'ayant pas de solutions calculables en temps raisonnable de façon analytique ou algorithmique.

En effet, des milliers de solutions (génotypes) plus ou moins bonnes sont créés au hasard puis sont soumises à un procédé d'évaluation de la pertinence de la solution mimant l'évolution des espèces : les plus "adaptés", c'est-à-dire les solutions au problème qui sont les plus optimales survivent davantage que celles qui le sont moins et la population évolue par générations successives en croisant les meilleures solutions entre elles et en les faisant muter, puis en relançant ce procédé .un certain nombre de fois afin d'essayer de tendre vers la solution optimale.

Par analogie à l'évolution génétique naturelle, dans un algorithme génétique, un individu (une solution) est caractérisé par une structure de données qui représente son empreinte génétique qu'on appelle le chromosome. Ce chromosome contient un ensemble de codes appelés gènes qui représentent les variables d'optimisation et qui peuvent prendre plusieurs valeurs appelées allèles [10]. La concaténation de tous les gènes de l'individu donne le chromosome qui lui représente une solution dans son intégralité.

[7]En appliquant l'algorithme, le mécanisme d'évolution et celui de sélection des individus sont inchangés entre générations, seules changent trois fonctions :

• la fonction qui s'occupe de représenter le problème en codant chaque information caractérisant une solution possible selon un codage bien particulier,

• la fonction inverse qui à partir d'un chromosome permet d'obtenir une solution par décodage du génome.

• la fonction qui évalue l'adaptation d'une solution à un problème, sa pertinence.

En d'autres termes, quand on utilise les algorithmes génétiques, aucune connaissance de la manière dont résoudre le problème n'est requise, il est seulement nécessaire de fournir une fonction permettant de coder une solution sous forme de gènes (et donc de faire le travail inverse) ainsi que de fournir une fonction permettant d'évaluer la pertinence d'une solution au problème donné.

Cela en fait donc un modèle minimal et canonique pour n'importe quel système évolutionnaire et pour n'importe quel problème pouvant être abordé sous ce paradigme.

3.2 Principales caractéristiques :

Les principales caractéristiques des algorithmes génétiques sont les suivantes :

• On utilise un codage des informations : on représente toutes les caractéristiques d'une solution par un chromosome (un ensemble de gènes) , sous un certain codage (binaire, réel, code de Gray, etc.), valeurs qu'on concatène pour obtenir une chaîne de caractères qui est spécifique à une solution bien particulière (il y a une bijection entre la solution et sa représentation codée)

• On traite une population "d'individus", de solutions : cela introduit donc du parallélisme.

• L'évaluation de l'optimalité du système n'est pas dépendante vis-à-vis du domaine.

• On utilise des règles probabilistes : il n'y a pas d'énumération de l'espace de recherche, on en explore une certaine partie en étant guidé par un semi-hasard car la fonction d'évaluation permet de choisir ,de s'intéresser à une solution qui semble représenter un optimum local, on fait donc un choix délibéré, puis de la croiser avec une autre solution optimale localement, En général la solution obtenue par croisement est meilleure ou du même niveau que ses parents, mais ce n'est pas assuré, cela dépend des aléas du hasard, et cela et d'autant plus vrai pour l'opérateur de mutation qui ne s'applique qu'avec une certaine probabilité et dans le cas où il s'applique choisit aléatoirement sur quel(s) locus introduire des modifications.[8]

3.3 Phases de l'algorithme génétique :

Le pseudo-code suivant représente les phases d'un algorithme génétique générique :

1) Initialiser la population initiale P.
2) Evaluer P.
3) TantQue (Pas Convergence) faire :
    a) P ' = Sélection des Parents dans P
    b) P ' = Appliquer Opérateur de Croisement sur P '
    c) P ' = Appliquer Opérateur de Mutation sur P '
    d) P = Remplacer les Anciens de P par leurs Descendants de P '
    e) Evaluer P

FinTantQue

Le critère de convergence peut être de nature diverse, par exemple :
• Un taux minimum d'adaptation de la population au problème, qu'on désire atteindre
• Un certain temps de calcul à ne pas dépasser,
• Une combinaison de ces deux points [8] .

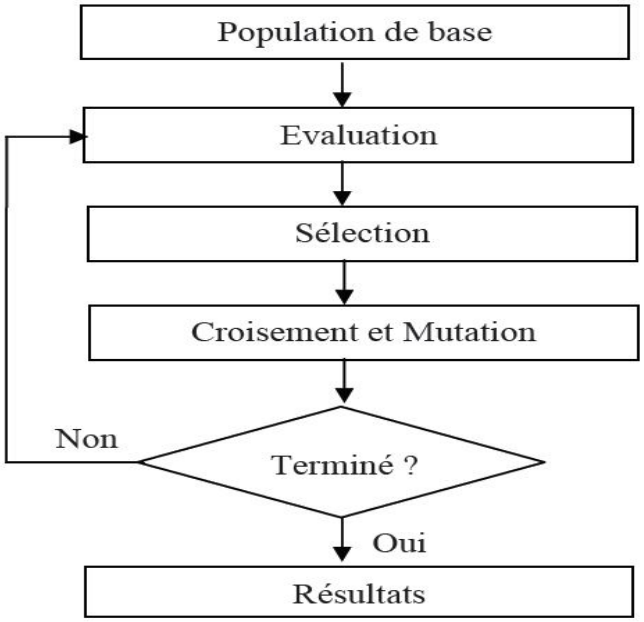

Figure 2 : Organigramme d'un Algorithme Génétique.

4. Configuration des paramètres et fonction de l'algorithme génétique :

4.1 . Codage et population initiale :

Le codage consiste à représenter les différents états possibles de la variable dont on cherche la valeur optimale sous forme utilisable. Cela permet d'établir une connexion entre la valeur de la variable et les individus de la population, de manière à imiter la transcription génotype-phénotype qui existe dans le monde vivant. Il existe principalement deux types de codage : le codage binaire, le codage réel [13] .

- ✓ **Codage binaire :**

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

- Les performances de l'algorithme sont dégradées devant les problèmes d'optimisation de grande dimension à haute précision numérique. Pour de tels problèmes, les AG basés sur les chaînes binaires ont de faibles performances comme le montre michalewicz[11].
- La distance de Hamming entre deux nombres voisins (nombre de bits différents) peut être assez grande dans le codage binaire : l'entier 7 correspond à la chaîne 0111 et la chaîne 1000 correspond à l'entier 8. Or la distance de hamming entre ces deux chaînes est de 4, ce qui crée bien souvent une convergence, et non pas l'obtention de la valeur optimale.

1	0	0	1

Figure 3:Représentation binaire.

✓ **Codage réel**

Il a le mérite d'être simple. Chaque chromosome est un vecteur dont les composantes sont les paramètres du processus d'optimisation. Par exemple, si on recherche l'optimum d'une fonction de n variables $f(x_1, x_2, x_{3,\ldots,} x_n)$, on peut utiliser tout simplement un chromosome Ch contenant les n variables : Avec ce type de codage, la procédure d'évaluation des chromosomes est plus rapide vu l'absence de l'étape de transcodage (du binaire vers le réel). Les résultats donnés par Michalewicz [11]montrent que la représentation réelle aboutit souvent à une meilleure précision et un gain important en termes de temps d'exécution [5] .

Figure 4: Représentation réel

4.2 Genèse aléatoire de la population initiale :

Le choix de la population initiale d'individus conditionne fortement la rapidité de l'algorithme :

Si la position de l'optimum dans l'espace d'état est totalement inconnue, il est naturel d'engendrer aléatoirement des individus en faisant des tirages uniformes dans chacun des domaines associés aux composantes de l'espace d'état, en veillant à ce que les individus produits respectent les contraintes [12].

Si par contre, des informations a priori sur le problème sont disponibles, il paraît bien évidemment naturel d'engendrer les individus dans un sous-domaine particulier afin d'accélérer la convergence. Dans l'hypothèse où la gestion des contraintes ne peut se faire directement, les contraintes sont généralement incluses dans le critère à optimiser sous forme de pénalités[13] .

4.3 Opérateurs de sélection :

Cet opérateur est chargé de définir quels seront les individus de P qui vont être dupliqués dans la nouvelle population P' et vont servir de parents lors du croisement.

Soit ***x*** le nombre d'individus de P, on doit en sélectionner ***x/2*** puisque l'opérateur de croisement nous permet de repasser à ***x*** individus.

Cet opérateur est peut-être le plus important puisqu'il permet aux individus d'une population de survivre, de se reproduire ou de mourir. En règle générale, la probabilité de survie d'un individu sera directement reliée à son efficacité relative au sein de la population.[8]

On reporte [8], quatre types de méthodes de sélection différentes,

- La sélection par Roulette Wheel

- La méthode "élitiste",
- La sélection par tournois,
- La sélection universelle stochastique.

4.2.1 La sélection par Roulette Wheel :

C'est la méthode la plus connue et la plus utilisée. La chance pour un individu d'être sélectionné est proportionnelle à sa performance, donc plus les individus sont adaptés au problème, plus ils ont de chances d'être sélectionnés.
Pour utiliser l'image de la "roue du forain", chaque individu se voit attribué un secteur dont l'angle est proportionnel à son adaptation, sa "fitness".
On fait tourner la roue et quand elle cesse de tourner on sélectionne l'individu correspondant au secteur désigné par une sorte de "curseur". La figure ci-dessous indique l'individu sélectionné quand le problème considéré est celui de minimiser la fonction objectif.

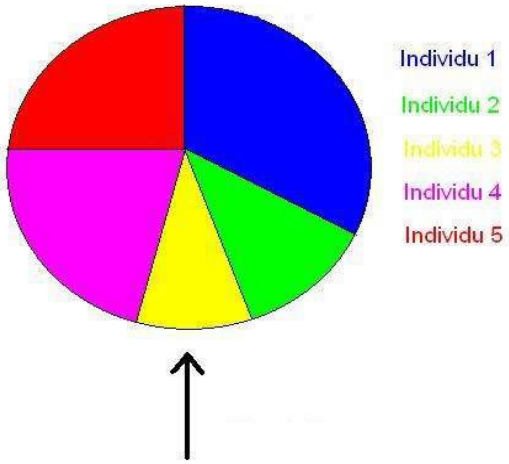

Figure 5 : Méthode de sélection de Roulette Wheel

4.2.2La méthode Elitiste :

Cette méthode consiste à sélectionner les **x** individus dont on a besoin pour la nouvelle génération **P'** en prenant les **x** meilleurs individus de la population **P** après l'avoir triée de manière décroissante selon le fitness de ses individus.
Notons que cette méthode est considérée pire que celle de la loterie biaisée dans le sens où elle amènera à une convergence prématurée encore plus rapidement et surtout de manière encore plus sûre que la méthode de sélection de la loterie biaisée : en effet, la pression de la sélection est trop forte, la variance nulle et la diversité inexistante. Le peu de diversité qu'il pourrait y avoir ne résultera pas de la sélection mais plutôt du croisement et des mutations.

4.2.3 La sélection par Tournois :

Selon, c'est avec cette méthode qu'on obtient les résultats les plus satisfaisants. Le principe de cette méthode est le suivant : on effectue un tirage avec remise de deux individus de P, et on les fait "combattre". Celui qui a le fitness la plus élevée l'emporte avec une probabilité p comprise entre 0.5 et 1. On répète ce processus n fois de manière à obtenir les n individus de P' qui serviront de parents.

La variance de cette méthode est élevée et le fait d'augmenter ou de diminuer la valeur de *p* permet respectivement de diminuer ou d'augmenter la pression de la sélection.[15]

4.2.4 La sélection universelle stochastique :

Cette méthode semble être très peu utilisée et qui plus est possède une variance faible, donc introduit peu de diversité, c'est pourquoi on se contentera de ce qui suit : On prend l'image d'un segment découpé en autant de sous-segments qu'il y a d'individus. Les individus sélectionnés sont désignés par un ensemble de points équidistants.[15]

4.3 Opérateurs de croisement :

La naissance d'un nouvel individu, nécessite la prise aléatoire d'une partie des gènes de chacun des deux parents. Ce phénomène, issu de la nature est appelé croisement (crossover). Il s'agit d'un processus essentiel pour explorer l'espace des solutions possibles. Une fois la sélection terminée, les individus sont aléatoirement répartis en couples. Les chromosomes parents sont alors copiés et recombinés afin de produire chacun deux descendants ayant des caractéristiques issues des deux parents. Dans le but de garder quelques individus parents dans la prochaine population, on associe à l'algorithme génétique une probabilité de croisement, qui permet de décider si les parents seront croisés entre eux ou s'ils seront tout simplement recopiés dans la population suivante. [9]

Il existe plusieurs méthodes de croisement comme par exemple: Le croisement en un point, Le croisement en multiples points, Croisement uniforme.

Croisement en Un-Point :

C'est le croisement le plus simple et le plus connu. Il consiste à choisir aléatoirement un point de croisement pour chaque couple de chromosomes. Les sous-chaînes situées après ce point sont par la suite inter-changées pour former les deux fils

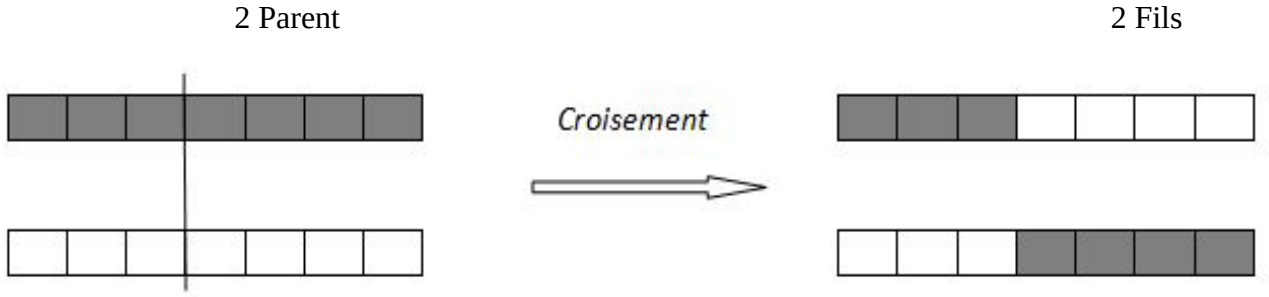

Figure 6: Croisement en Un Point

Croisement en n-points :

Ce type de croisement s'énonce par un choix aléatoire de **n**-points de coupure pour dissocier chaque parent en **n+1** fragments. Pour former un fils, il suffit de concaténer alternativement **n+1** sous chaînes à partir des deux parents. Ce croisement cherche à explorer tout l'espace de solutions possibles en créant des descendants ayant des caractéristiques très loin des parents. [15]

On choisit aléatoirement deux points de coupure (n=2):

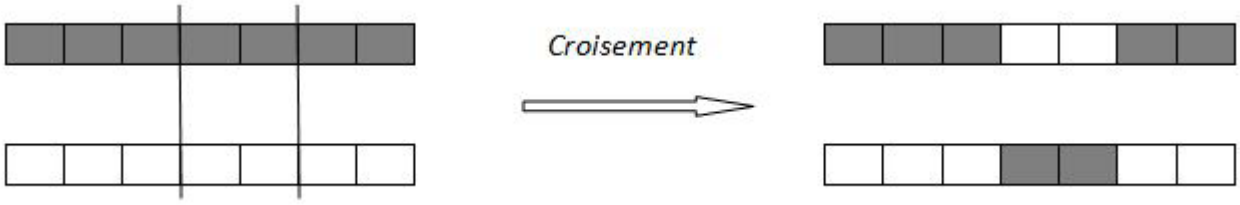

Figure 7: Croisement en 2 point.

Croisement uniforme :

Cette technique génère des progénitures gène par gène à partir des deux parents. Il existe des versions distinctes de ce croisement. La plus connue est celle qui utilise un masque. S'il est égal à 1, l'enfant 1 reçoit l'allèle correspondant du parent 1 et l'enfant 2 reçoit celui du parent 2. Sinon, l'échange se fait dans l'autre sens. [9

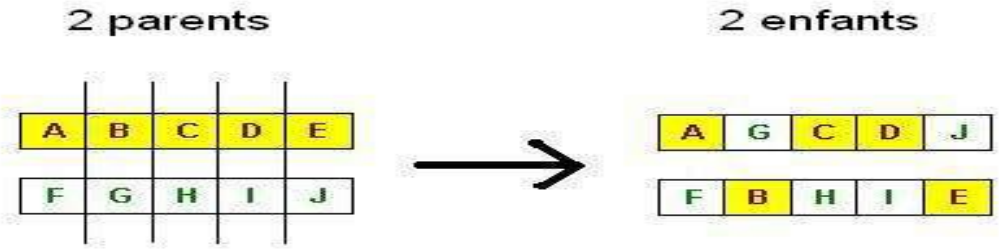

Figure 8: Croisement uniforme.

4.4 Opérateur de Mutation :

L'opérateur de mutation consiste à changer la valeur allélique d'un gène avec une probabilité p_m très faible, généralement comprise entre 0.01 et 0.001.

L'opérateur de mutation modifie donc de manière complètement aléatoire les caractéristiques d'une solution, ce qui permet d'introduire et de maintenir la diversité au sein de notre population de solutions. Cet opérateur joue le rôle d'un "élément perturbateur", il introduit du "bruit" au sein de la population. [14]

- **Exemple :**

En codage binaire, une mutation consiste simplement en l'inversion d'un bit (ou de plusieurs bits, mais vu la probabilité de mutation c'est extrêmement rare). Le bit en question est déterminé de manière aléatoire [14];

Ci-après, un pseudo-code pour représenter la mutation, où on utilise une fonction censée nous retourner **true** avec une probabilité p_m.

Pour chaque bit **faire**
appel à la fonction de mutation
Si cette fonction nous renvoie *true* **alors**
on inverse le bit
FinSi
FinPour

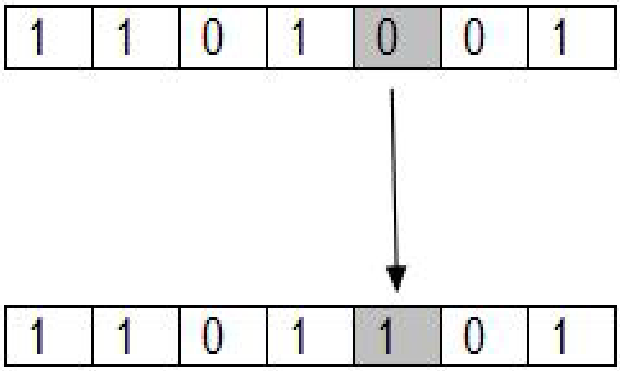

Figure 9: Une Mutation

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. [13]

5 Exemple de mise en œuvre :

L'exemple suivant, tiré de [Goldberg], va permettre de mettre en œuvre, sur une génération, tous les opérateurs étudiés de manière à pouvoir observer le résultat de l'application successive de ces opérateurs.

Il consiste à trouver le maximum de la fonction "f (x)= x" sur l'intervalle [0;31] où x est un entier. La première étape consiste à coder la fonction. Par exemple, nous utilisons un codage binaire de x,

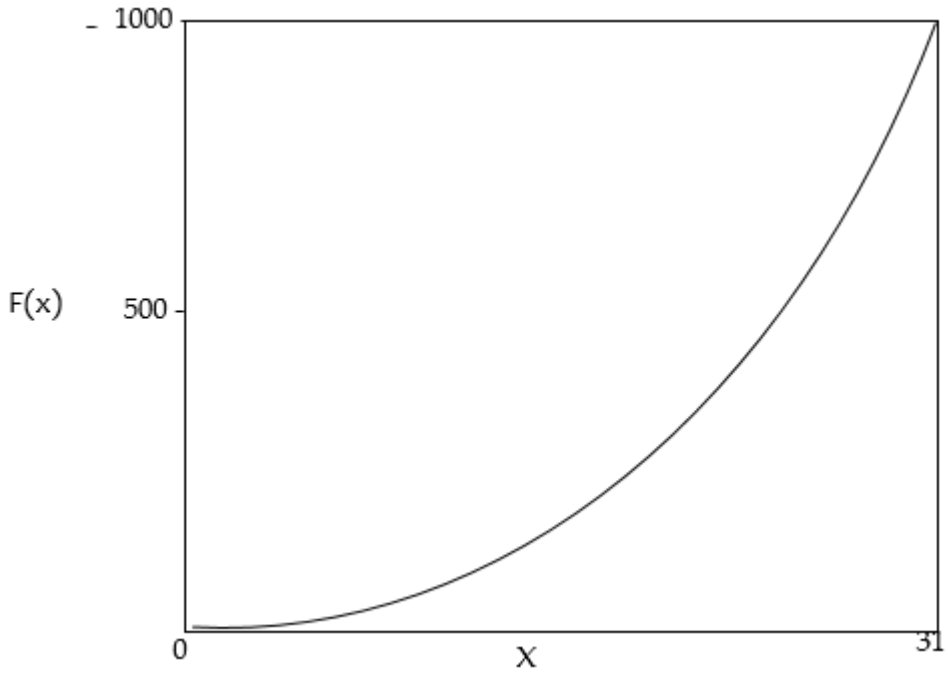

Figure 10 : Exemple simple d'optimisation de fonction,

La fonction $f(x)=x^2$ sur l'intervalle entier [0, 31] [godelberg]

On à 32 valeurs possibles pour *x* on choisit donc un codage discret sur 5 bits. Ainsi, nous avons

x = 2 {0, 0, 0, 1,0}, de même x = 31 {1, 1, 1, 1,1}. Nous recherchons donc le maximum d'une fonction de fitness

5.1 Tirage et évaluation de la population initiale :

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

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

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

Tableau 1: Principe de fonctionnement d'un algorithme génétique

5.2 Sélection :

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

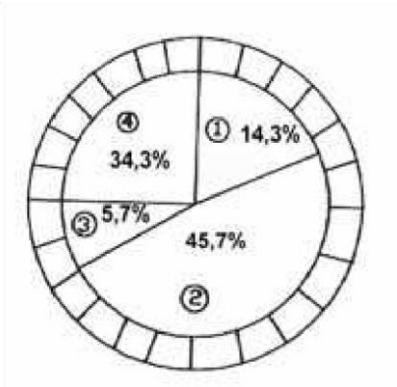

Figure 11: La Roue De Roulette Wheel : Opération De Sélection

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

Numéro	Séquence
1	10000
2	01100
3	00101
4	10000

Tableau 2: Sélection par Roulette Wheel

5.3 Le Croisement :

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

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

Tableau 3: Principe du crossover

5.4 La Mutation :

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

Chromosome	Tirage Aléatoire	Nouveau Bit	Nouveau Chromosome
10001	15 25 36 04 12	1	10011
00100	26 89 13 48 59	-	00100
01000	32 45 87 22 65	-	01000
10100	47 01 85 62 35	1	11100

Tableau 4: Mutation des chromosomes

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

5.5 Retour à la phase d'évaluation :

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

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

Tableau 5: Nouvelle génération

6 . Conclusion :

Les algorithmes génétiques peuvent être particulièrement utiles dans les domaines suivants : optimisation de fonctions, des emplois du temps(planification), l'apprentissage des réseaux de neurones pour la classification, la prédiction, la robotique, ou en étude du vivant et du monde réel : marchés économiques, systèmes industriels, comportements sociaux, systèmes immunitaires, etc...

Ils apportent assez rapidement une solution acceptable. Néanmoins, il est possible de l'améliorer assez efficacement en le combinant avec un algorithme déterministe.[14]

CHAPITRE 3: Outils de simulation du trafic routier

1. Introduction :

Ce chapitre expose quelques simulateurs du trafic routier. Nous donnons une vue générale sur leurs caractéristiques fondamentales et leurs fonctionnalités. Dans la première partie, on s'intéresse d'avantage au simulateur utilisé dans notre travail : SUMO. La deuxième partie présente deux autres simulateurs pour le trafic routier.

2. Le simulateur SUMO :

2.1 Caractéristiques principales :

SUMO (Simulation of Urban Mobility) est un logiciel de simulation du trafic routier, open source sous licence ONU public (OPL), dont le développement a commencé 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 est 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. [10]

SUMO est conçu pour des simulations microscopiques du trafic routier et continues dans l'espace : il s'intéresse aux caractéristiques spatio-temporelles aussi bien de chaque véhicule individuellement qu'à celles entre véhicules également. Les caractéristiques des infrastructures routières et leurs influences sur le trafic sont également prises en considération à un niveau microscopique (relativement à chaque véhicule individuellement). Cette modélisation individuelle des véhicules et de leurs interactions est réalisée à l'aide de modèles mathématiques de comportement en matière de suivi de voiture, de changement de voie et d'intersection.

Le déploiement du paquet SUMO requiert l'installation des bibliothèques C++ uniquement, i.e. il est très portable.

2.2 Eléments des scénarios en SUMO :

SUMO offre aux utilisateurs le moyen d'insérer une large gamme de variantes du trafic routier et donc d'aborder un large éventail de sujets de gestion du trafic :

- Prend en charge le trafic terrestre multimodal et intermodal.
- Il utilise également des modèles piétonniers pour simuler le mouvement des personnes et leurs interactions avec les véhicules [16]
- Supporte la description des véhicules au plus fin détail (vitesse, accélération, taille, forme,…etc

- Décrit le déplacement des véhicules avec précision, en termes de date et lieu de départ d'arrivée, de parcours, d'attente, de variation de vitesse, le changement de voie …etc
- Permet la description des parcours au niveau flux de véhicules ou au niveau des véhicules individuellement.
- La création de l'infrastructure routière via différentes méthodes : par insertion personnalisée des composantes ou par importation des cartes routières satellitaires en .osm ou en d'autres formats.
- Assure l'insertion et le respect des règles de circulation, de priorité selon les codes de la route (quand le scénario ne demande pas le contraire), tant au niveau des routes, des intersections, des feux, …etc

2.3 Composition des simulations :

La création des scénarios est réalisé à la base via du code XML ce qui permet d'apporter la précision voulue au trafic routier étudié. Néanmoins, SUMO offre des outils et interfaces graphiques afin de simplifier cette tâche et de regrouper les différents composants.

2.3 .1. Composants des scénarios en XML :

a. Généralités :

Pour créer une configuration de scénario SUMO, il est nécessaire d'utiliser un fichier de définition de schéma XML.

Les éléments XML et les attributs sont regroupés en clauses. Leurs valeurs sont de types variables :

<BOOL> : une valeur booléenne, utilisez "t" ou "true" et "f" ou "false" pour le codage
<INT> : une valeur entière, peut être négative
<UINT> : un entier non signé, doit être> = 0
<FLOAT> : un nombre à virgule flottante
<TIME> : heure, exprimée en secondes ; les fractions sont autorisées, par exemple "12.1 "
<STRING> : n'importe quelle chaîne, mais utilisez uniquement des caractères ASCII
<ID> : une chaîne qui ne doit pas contenir les caractères suivants : '#'

b. Utilisation des fichiers de scénarios :

les fichiers XML à intégrer à la simulation sont principalement :

< NETWORK_FILE> : un fichier réseau SUMO construit par les outils fournis NETGENERATE ou NETCONVERT

< ROUTES_FILE> : un fichier des routes SUMO tel que construit par DUAROUTER ou JTRROUTER ou manuellement

< TYPE_FILE> : un fichier de type SUMO Edge construit à la main ou téléchargé

< OSM_FILE> : un fichier OpenStreetMap tel qu'exporté depuis OpenStreetMap+

Le tableau ci-après montre quelques options à ajouter sur le fichier de configuration pour insérer les fichiers référencés :

Option	Description
-n <FILE> **--net-file <FILE>**	Charger la description du réseau routier à partir du fichier
-r <FILE> **--route-files <FILE>**	Charger les descriptions d'itinéraires à partir de FICHIER (s)
-a <FILE> **--additional-files <FILE>**	Charger d'autres descriptions à partir de FICHIER (s)
-w <FILE> **--weight-files <FILE>**	Charger d'autres descriptions à partir de FICHIER (s)
-x <STRING> **--weight-attribute <STRING>**	Nom de l'attribut xml qui donne le poids de l'arête; défaut: traveltime
--load-state <FILE>	Charge un état de réseau à partir de FILE
--load-state.offset <TIME>	Décale tous les temps chargés d'un état sauvegardé du décalage donné; défaut: 0
--load-state.remove-vehicles <STRING>	Supprime les véhicules avec les identifiants donnés de l'état chargé

Tableau 6: Input [18]

2.3 .2 Applications du paquet SUMO :

SUMO comprend plusieurs applications utilisables pour créer les éléments de la simulation. On cite parmi tant d'autres :

Nom de l'application	Description courte
SUMO	La simulation microscopique sans visualisation; application en ligne de commande
SUMO-GUI	La simulation microscopique avec une interface utilisateur graphique
SUMO-CONSOLE	La simulation microscopique mode console
NETCONVERT	importateur et générateur de réseau; lit les réseaux routiers de différents formats et les convertit au format SUMO
NETEDIT	Un éditeur de réseau graphique.
NETGENERATE	Génère des réseaux abstraits pour la simulation SUMO

Tableau 7: quelques applications disponibles en SUMO [17]

La Figure ci-après montre l'interface principale de SUMO-GUI. Elle présente une suite de menus et commandes qui assistent à préparer et à exécuter la simulation d'un scénario de trafic routier.

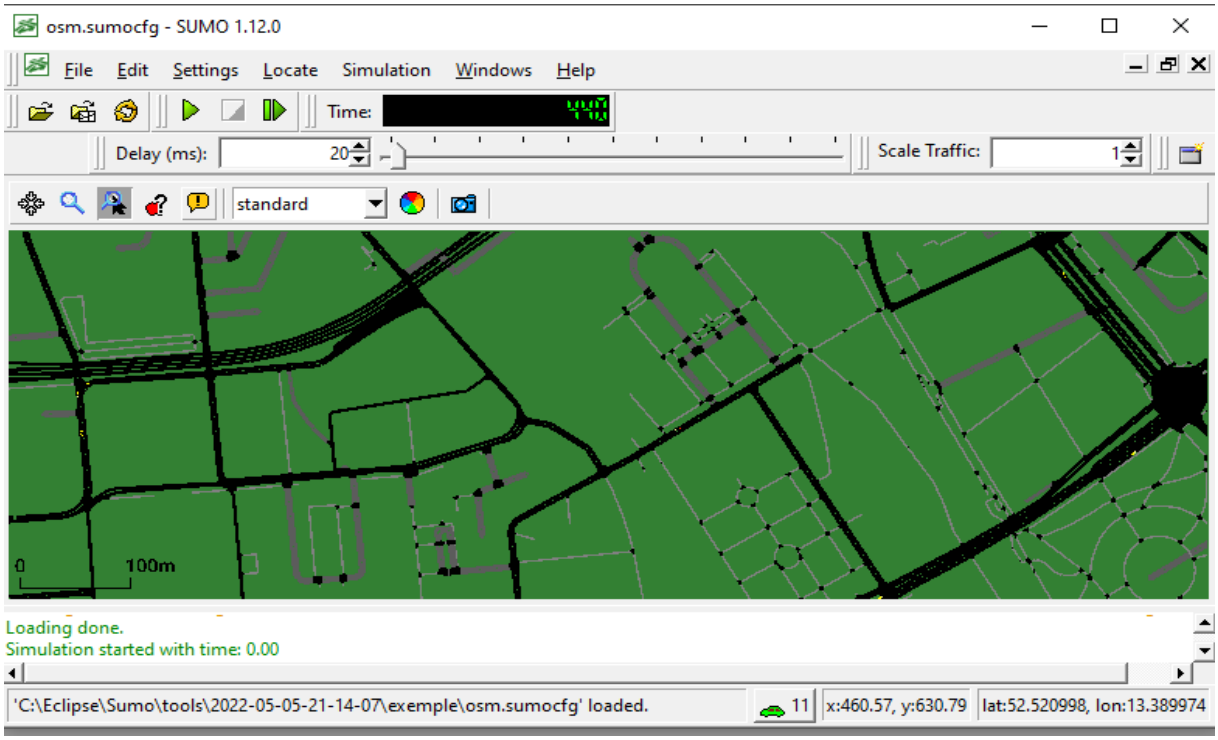

Figure 12: interface principale d'une simulation en Sumo

2.3 .3 Traces des simulations :

L'objectif principal de SUMO n'est pas uniquement de produire des scénarios aussi réalistes que possibles mais c'est aussi de produire le maximum de résultats fiables, précis et utilisables sans pour autant alourdir le déroulement des simulations. La table suivante montre quelques exemples des traces récupérables à la suite des simulations et les options pour les demander.

Option	Description
--write-license <BOOL>	Inclure les informations de licence dans chaque fichier de sortie; défaut: false
--output-prefix <STRING>	Préfixe appliqué à tous les fichiers de sortie. La chaîne spéciale 'TIME' est remplacée par l'heure actuelle.
--precision <INT>	Définit le nombre de chiffres après la virgule pour la sortie en virgule flottante; défaut: 2
--precision.geo <INT>	Définit le nombre de chiffres après la virgule pour les sorties lon, lat; défaut: 6
-H <BOOL> **--human-readable-time <BOOL>**	Ecrivez les valeurs horaires en heure: minute: seconde ou jour: heure: minute: seconde plutôt que seconde; défaut: false
--netstate-dump <FILE>	Sauvegarder les états complets du réseau dans FILE
--netstate-dump.empty-edges <BOOL>	Écrivez également les bords complètement vides lors du déchargement; défaut: false
--netstate-dump.precision <INT>	Ecrire les positions et les vitesses avec la précision donnée (défaut 2); défaut: 2
--emission-output <FILE>	Enregistrer les valeurs d'émission de chaque véhicule
--emission-output.precision <INT>	Ecrire les valeurs d'émission avec la précision donnée (par défaut 2); défaut: 2

Tableau 8 : traces des simulations [18]

2.3 .4 Autres paramètres de configurations des scénarios :

a. Timing des simulations :

Le tableau suivant indique quelques options pour contrôler le timing des simulations :

Option	Description
-b <TIME> **--begin <TIME>**	Définit le temps de début en secondes; La simulation commence à ce moment; défaut: 0
-e <TIME> **--end <TIME>**	Définit l'heure de fin en secondes; La simulation se termine à ce moment; défaut: -1
--step-length <TIME>	Définit la durée du pas en secondes; défaut: 1

Tableau 9 : options de contrôle des temps de simulation [18]

b. Déroulement graphique des simulations

Option	Description
-g <FILE> **--gui-settings-file <FILE>**	Charger les paramètres de visualisation depuis le fichier
-Q <BOOL> **--quit-on-end <BOOL>**	Quitte l'interface graphique lorsque la simulation s'arrête. défaut: false
-G <BOOL> **--game <BOOL>**	Démarrez l'interface graphique en mode jeu. défaut: false
--game.mode <STRING>	Sélectionnez le type de jeu ('tls', 'drt'); défaut: tls
-S <BOOL> **--start <BOOL>**	Lancer la simulation après le chargement ; défaut : false
-B <STRING> **--breakpoints <STRING>**	Utilisez TIME [] comme moments où la simulation doit s'arrêter
--edgedata-files <FILE>	Poids de la rive / de la voie pour la visualisation à partir de FILE
-D <BOOL> **--demo <BOOL>**	Redémarrez la simulation après avoir terminé (mode démo); défaut: false
-T <BOOL> **--disable-textures <BOOL>**	Ne chargez pas d'images d'arrière-plan; défaut: false
--registry-viewport <BOOL>	Charger la fenêtre d'affichage actuelle à partir du registre; défaut: false
--window-size <STRING>	Créer une fenêtre initiale avec la taille x, y donnée
--window-pos <STRING>	Créer la fenêtre initiale à la position x, y donnée
--tracker-interval <FLOAT>	La période d'agrégation pour les fenêtres de suivi de valeur; défaut: 1
--osg-view <BOOL>	Commencez avec une vue OpenSceneGraph au lieu de la vue 2D normale; défaut: false
--gui-testing <BOOL>	Activer le délai d'attente pour la reconnaissance d'écran; défaut: false
--gui-testing-debug <BOOL>	Activer les messages de sortie lors du test de l'interface graphique; défaut: false

Tableau 10 : options pour l'application SUMO-GUI [18]

c. Modifications supplémentaires aux simulations :

Quant au tableau suivant, il présente quelques-unes des options pour paramétrer ou modifier des détails de la simulation sans modifier les fichiers XML de départ.

Option	Description
--no-internal-links <BOOL>	Désactiver (jonction) les liens internes; défaut: false
--threads <INT>	Définit le nombre de threads pour la simulation parallèle. défaut: 1
--lateral-resolution <FLOAT>	Définit la résolution en m lors du traitement du positionnement latéral dans une voie (-1 étant tous les véhicules conduisant au centre de leur voie; par défaut: -1
-s <HEURE> **--route-steps <TIME>**	Charge les itinéraires pour le nombre de secondes à venir; défaut: 200

Tableau 11 : options pour traitement des détails supplémentaires [18]

2.4 TraCi :

TraCI signifie «Traffic Control Interface». Il s'agit d'une interface qui consiste en un ensemble de commandes qui permettent à l'utilisateur d'avoir accès à la simulation du trafic en cours d'exécution. Il est donc possible de récupérer des valeurs d'objets simulés et de manipuler leur comportement en ligne avant la fin de la simulation. TraCI peut être utilisé soit comme interface pour d'autres programmes, soit au moyen de scripts python qui interagissent avec la simulation. [19]

2.4.1. Utiliser TraCI :

Démarrage de SUMO : l'interfaceTraCI utilise une architecture client / serveur basée sur TCP pour fournir un accès à SUMO. Ainsi, l'application SUMO agit en tant que serveur démarré avec des options de ligne de commande supplémentaires : --remote-port ou est le port que SUMO écoutera pour les connexions entrantes.

Lorsqu'il est lancé avec l'option --remote-port , SUMO prépare uniquement la simulation et attend que toutes les configurations des scénarios se connectent et prennent le contrôle. Veuillez noter que l'option --end est ignorée lorsque SUMO s'exécute en tant que serveur TraCI. SUMO s'exécute alors jusqu'à ce que le client demande la fin de la simulation.[20]

Lors de l'utilisation de SUMO-GUI en tant que serveur, la simulation doit être lancée à l'aide du bouton de lecture ou en définissant l'option --start avant que les commandes TraCI ne soient traitées.

Notons que même si le principe du contrôle des simulations fait intervenir le protocole TCP, son utilisation effective est simple et repose sur des commandes compréhensibles et facilement utilisables.

2.4.2. Plusieurs simulations à la fois :

Le nombre de clients pouvant se connecter peut-être indiqué en option supplémentaire --num-clients , où 1 correspond à la valeur par défaut. Veuillez noter que dans les scénarios multi-clients, il faut spécifier explicitement l'ordre d'exécution des clients à l'aide de la commande SetOrder.

Chaque client doit spécifier une valeur entière unique, généralement arbitraire, et les commandes du client seront traitées dans l'ordre, de la valeur la plus basse à la valeur la plus élevée dans chaque étape de la simulation.

Les clients sont automatiquement synchronisés après chaque étape de la simulation. Cela signifie que la simulation n'avance pas à l'étape suivante tant que tous les clients n'ont pas appelé la commande 'simulationStep'. De plus, la commande simulationStep ne restitue le contrôle au client qu'une fois la simulation avancée. [20]

3.Le simulateur VISSIM :

VISSIM est un logiciel de simulation microscopique de la circulation. Développé pour modéliser la circulation en milieu urbain et les transports en commun, il repose sur un modèle psycho-physique du conducteur développé par R. Wiedemann . Ce logiciel est un outil puissant et complexe permettant l'évaluation efficace d'aménagement de la circulation. VISSIM est un logiciel populaire et reconnu, utilisé dans de nombreuses études en recherche et dans la pratique.

Les tronçons et les liaisons sont les éléments de base utilisés pour représenter le réseau routier. Les tronçons représentent typiquement les sections de route. Les liaisons représentent aussi des sections de route, et sont typiquement utilisés pour les mouvements tournants et la connexion des tronçons. Un tronçon est caractérisé par un nombre de voies, donc il faut créer des tronçons distincts pour représenter des routes avec différents nombres de voies.

Un modèle de simulation de la circulation dans VISSIM correct n'a pas de collision, mais des véhicules sur des tronçons et liaisons différents ne se voient pas (comme s'ils étaient à des hauteurs différentes) ! d'où des erreurs de visualisation lorsque les véhicules semblent se passer les uns sur les autres dans les animations. [21]

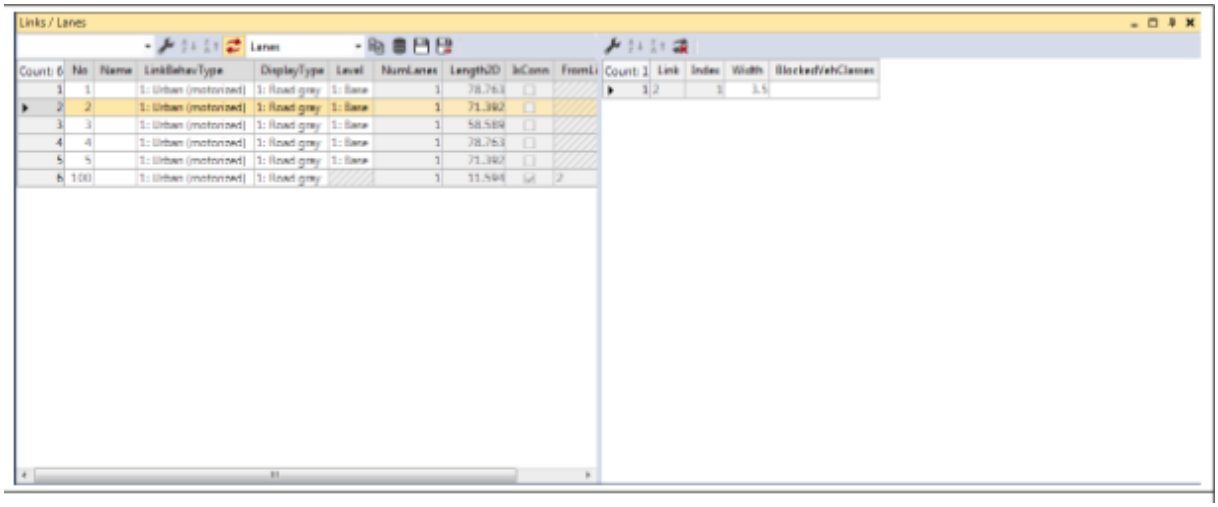

Figure 13 : Liste des tronçons et liaisons dans un réseau[21]

4. Le simulateur AnyLogic :

AnyLogic est un outil de simulation développé par The AnyLogic Company. Il possède un langage de modélisation graphique et facilite également l'extension du modèle de simulation avec le code Java. [22]

La modélisation de simulation AnyLogic fournit une Bibliothèque de trafic routier, qui offre la flexibilité et la puissance nécessaire pour parvenir à une ingénierie et à une conception du trafic routier plus efficaces. Une visualisation claire est un moyen simple et rapide pour faciliter le développement de modèles d'infrastructures routières, avec des cartes de densité présentant la congestion et des animations montrant le flux du trafic et les goulets d'étranglement. La liberté de pouvoir faire des expériences et la capacité à optimiser des modèles précis sont les points forts de cet outil. [23]

AnyLogic fournit un ensemble exceptionnel d'outils spécifiques à un secteur dans un seul package, sans frais supplémentaires. Il présente les bibliothèques suivantes :

une **bibliothèque de modélisation de processus** pour les processus ou flux de travail d'entreprise génériques.

La **bibliothèque de fluide** permet de simuler les cargaisons en vrac et le transport de liquides dans des industries telles que l'industrie minière ou pétrolière et gazière.

Quant à la **bibliothèque ferroviaire** pour le transport ferroviaire, les terminaux et le triage.

Bibliothèque piétons pour les flux de piétons dans les aéroports, les stades, les gares ou les centres commerciaux.

Bibliothèque de trafic routier pour le mouvement des voitures, camions et bus sur les routes, les parkings et les sites de production.

Bibliothèque de manipulation de matériaux pour les processus de fabrication et de manutention.

Les Bibliothèques piétons, ferroviaires et de trafic routier offrent une simulation détaillée au niveau physique des mouvements et interactions des objets, qui n'est pas prise en charge par les autres outils logiciels de simulation à visée générale. [24]

5. Conclusion :

Dans ce chapitre nous avons présentés trois simulateurs existants pour le trafic routier. Parmi ces simulateurs, on s'est intéressé davantage à SUMO vu que nous l'utilisons dans notre travail.

1. Introduction :

Dans ce chapitre, Nous décrivons la mise en œuvre de la technique utilisée en justifiant certains choix de configuration de l'algorithme d'optimisation et de l'environnement de simulation utilisés. Nous présentons les détails de développement de l'algorithme génétique déployé pour le contrôle des feux de signalisation en zone urbaine simulé en SUMO.

2. Contextes du contrôle proposé :

Les algorithmes génétiques sont utilisés pour calculer des solutions optimales dans les problèmes de différents domaines. Le domaine du contrôle du trafic routier en fait cas de figure et les résultats rapportés sont très intéressant et encourageant à l'utiliser.

En effet, d'un côté, l'algorithme génétique est très simple et peut être facilement compris et mis en œuvre. Il lance la recherche d'une solution optimale parmi un groupe de solutions et non une solution unique. De plus, il utilise des règles probabilistes pour trouver la solution optimale et non pas des règles déterministes, qui offrent davantage de diversité.

D'un autre côté, même si le nombre de travaux effectués en utilisant l'algorithme génétique pour le trafic routier est important, aucun d'eux n'est effectué en Algérie et encore moins sur le réseau routier urbain de Mila.

Ces constats nous mènent à explorer la faisabilité et la performance de cet algorithme dans le contexte du contrôle des feux de signalisation appliqué carrefour urbain de la wilaya de Mila et ceux dans le cadre d'un projet de fin d'études en Master informatique.

La nécessité et l'importance d'une telle étude est validée sans contestation vue les mauvaises conditions de circulation subies aux quotidiens aussi bien par les conducteurs dans le cadre sociale qu'économique.

L'objectif de notre travail est de proposer une solution au problème de régulation du trafic via feux de signalisation. Nous déployons cette solution sur le réseau routier du centre-ville de MILA sur les feux tricolores en plusieurs point d'intersection de l'infrastructure routière. Cette proposition est basée sur l'algorithme génétique et testée sur le simulateur SUMO.

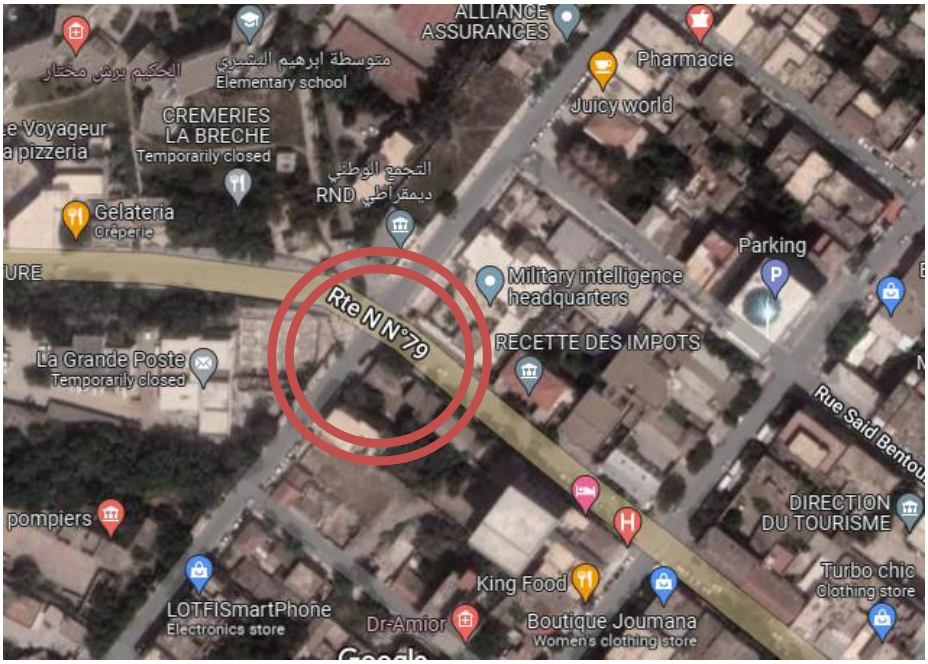

Figure14 : Image OSM par satellite de l'intersection étudier

3. Problème des feux de circulation :

Les feux de signalisation routière dans la zone traitée fonctionnent suivant un système rudimentaire. Quand le feu rouge s'allume, les voitures doivent s'arrêter durant toute la durée du feu qui est de 30 secondes, même si la voie opposée est libre. Cela favorise l'immobilité des véhicules qui est génératrice d'une multitude de problèmes : embouteillage, perte de temps et de carburant pour les automobilistes, pollution par le gaz d'échappement, stress et nuisances sonores. Par contre, si l'on considère la durée du feu vert, qui est synonyme de mobilité, elle n'est que de 28 secondes. Il est donc évident que le système en place n'est plus adapté à l'esprit actuelle, portée plutôt sur la fluidification de la circulation routière par la réduction des temps d'arrêt à travers une optimisation du fonctionnement des systèmes lumineux de la circulation.

4. Paramètres de l'algorithme génétique :

Nous présentons dans ce qui suit les principales phases et leurs valeurs choisies lors de la mise en exécution de l'algorithme génétique combiné à SUMO pour minimiser le temps d'attente

4.1. Codage des chromosomes :

L'étape de codage est l'une des étapes les plus importantes de l'algorithme génétique. Une solution candidate ou chromosome dans notre cas est une liste de X gènes. Chaque gène correspondant à une durée de phase d'un feu est représenté par un nombre entier, entre 10 et 30 secondes.

Les valeurs choisies entre 10s et 30s sont recommandées auprès des autorités des transports pour éviter des temps d'attentes trop longs devant les feux et de même assurer la sécurité lors du passage entre phases en marquant au moins 10s d'arrêt avant le redémarrage.

Le nombre X de gènes dépend du nombre de feux considérés et des phases de chaque feu. En considérant une seule intersection ayant 4 phases, comme l'exemple illustré sur la figure 15 ci-dessous, le nombre X de gènes vaut 4. Si d'autres intersections, munies de feu de signalisation, sont aussi prises en considération dans le contrôle, alors le nombre de phases de chaque feu est rajouté à X. La figure 5 montre le codage des chromosomes.

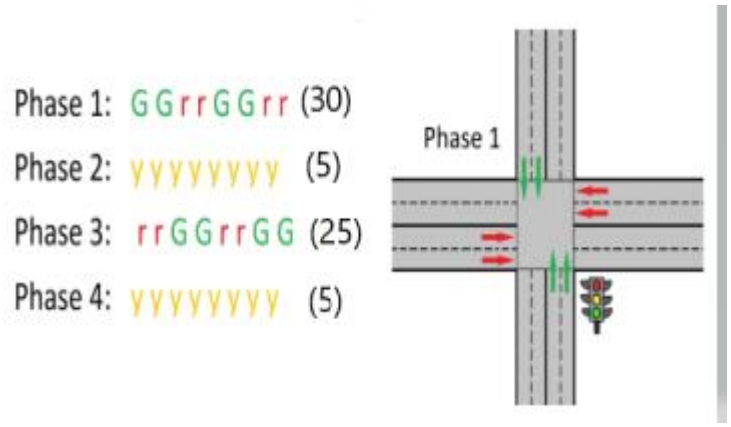

Figure 14: exemple des phases sur une intersection contrôlée par un feu de signalisation.

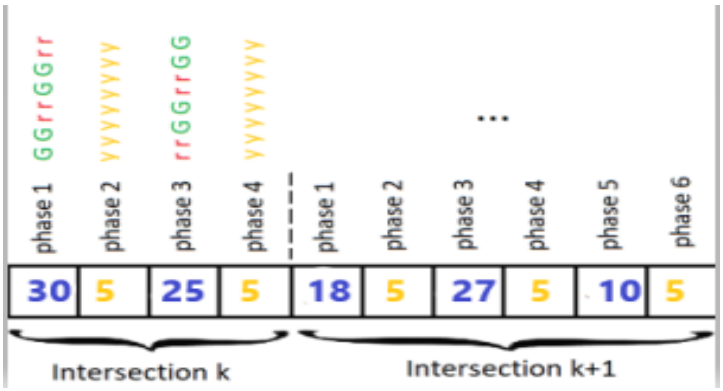

Figure 15 : codage d'un chromosome pour contrôler deux feux à 4 et 6 phases respectivement

Remarquons que les phases intermédiaires entre deux phases de circulation ont toujours une durée fixe et courte. En effet, ces phases sont insérées pour assurer la sécurité sur la route sans pour autant retarder la circulation. Dans les eux exemples illustrés, cette durée est de 3 secondes.

4.2. Création des premiers ancêtres des générations :

Après l'étape de codage, la création de la population est l'étape qui suit, où les premiers individus seront créés pour former la population initiale en respectant le codage établi. Ça sera la première population à être évaluée pour sélection et reproduction des générations.

Cette étape consiste à déterminer deux paramètres : quelles sont les valeurs pour les premiers gènes et quel nombre d'individus à créer. En termes d'optimisation, il s'agit de placer les éventuelles solutions initiales dans la zone de recherche selon une certaine densité qui ne ralentit pas le déroulement de l'algorithme génétique, i.e. éviter une recherche exhaustive sur le domaine des solutions.

Selon la littérature, il n'existe pas de paramétrage universel pour la quantification du dimensionnement. Cependant, certaines valeurs largement utilisées pour résoudre concrètement des problèmes sont retenues telle pour la taille de la population : entre N = 30 et 50 individus

Dans notre cas, la population initiale sera composée d'un nombre d'individus N=10 créé chacun avec des valeurs uniformément aléatoires. Ce choix est imposé par un compromis entre les capacités matérielles des ordinateurs utilisés et les délais de remise de nos résultats.

Les gènes des chromosomes initiaux codant une attribution initiale des temps de feux verts des différentes routes sur les intersections choisies, en respectant les contraintes de sécurité routières.

Le pseudo-code suivant résume les étapes de la création de la population initiale .on à utiliser langage Python

#initialisation

S=[] # population initiale = Ø

Initialisation de population**(**choisi aléatoire 4 valeur de l'intervalle [10,30] et ajouter l'individu de ces 4 gène a liste de population**).**

Répéter l'opération d'initialisation 10 fois et ajouter à chaque fois à la liste S.

#**Après passe à la partie d'évaluation de population**

Algorithme 4.1 : Etape de création de la population initiale

```
L'Initialisation de poulation S

[ (12, 5, 18, 5),  (12, 5, 18, 5), (27, 5, 27, 5),
(30, 5, 19, 5), (10, 5, 11, 5)),  (20, 5, 18, 5),
(21.43, (12, 5, 23, 5), (12, 5, 23, 5), (28, 5, 29, 5),
 (10, 5, 30, 5)]
```

Figure 17: L'affichage de l'initialisation de population

Après l'initialisation de population ,on à fait un évaluation des individus par Sumo, pour mettre à jour les phase de l'intersection étudier et pour calculer le temps d'attente moyen de cet intersection durant la durée de simulation .

Après le calcule de temps moyen on ajoute à la liste (RankedSolution) avec son individu **RankedSolution(MinWatingTime,(individu))** Chaque individu 0

4.3. Sélection des individus :

Après évaluation des individus, ils 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 élitiste.

Donc à travers la liste (RankedSolution) ,Nous trions le temps d'attente par ordre croissant **(RankedSolution.sort())** et choisissons les 4 première valeur de la liste avec son individu.

Les 4 individu représente les parent. Et voila on a fait la sélection de les meilleurs parent

4.4. Reproduction des générations :

A travers le même codage de la génération initiale, de nouvelles valeurs de gènes sont produites, selon un processus évolutionnaire, et permettant de s'adapter aux changements de la circulation sur la route et donc d'aboutir à un chromosome qui puisse optimiser le trafic routier.

4.4.1 Croisement :

Le croisement permet d'obtenir de nouveaux individus parmi les individus existants dans la population. En termes de recherche de solution optimale, il s'agit de visiter de nouveaux emplacements parmi les solutions potentiels.

Cette étape est principalement effectuée en choisissant deux individus de la population et en appliquant l'opérateur de croisement sur eux. Le résultat sera un ou deux nouveaux individus (le résultat est un individu si on obtient deux individus identiques).

Cependant, selon [26], il est difficile de trouver un type de croisement optimal. Il dépend du problème traité. Un opérateur de croisement universel (ou standard) n'existe pas et n'existera peut-être pas. Les auteurs de cette même référence assurent que dans les problèmes de grande taille, l'exécution de l'opérateur de croisement entre deux parents seulement pourrait être peu efficace. Il est donc beaucoup plus intéressant de procéder au croisement de plusieurs parents.

Nous avons suivi les recommandations de cette dernière étude, en créant les gènes des fils à partir de ceux de plusieurs parents choisis aléatoirement.

De plus, même si une solution largement utilisée est d'effectuer des croisements multipoints, il est tout à fait possible de faire des croisements aléatoires.

Le pseudo code suivant résume les étapes du croisement

4.4.2 Mutation :

La mutation consiste à altérer un gène dans un chromosome selon un facteur de mutation. Ce facteur est la probabilité qu'une mutation soit effectuée sur un individu.

Dans notre cas, nous avons fait la mutation en multipliant la valeur de 1er et 3eme case par une valeur aléatoire de l'intervalle [0.99 ,1.01] ,après ca générer le nouveau génération de 10 individu.

On a générer10 génération. Dans chaque génération on a fait la simulation et calculer le temps d'attente .

L'algorithme suivant montre les étapes de l'opérateur de croissement mutation.

Début

1- Sélectionner les premier 4 individus de 4 phase sans temps d'attente.
2- Ajouter les individu a la liste Elément [].
3- On fait la croissement par choisissez une valeur aléatoire dans la de la 1er cas 3eme case de n'importe
quel individu pour géré un enfant de 4 phase [e1,e2,e3 ,e4]
4- la valeur de 2eme et 4eme case est fixé e2=e4==5.
5- Appliquer l'opérateur de la mutation choisie
6- Générer 10 enfant a partir de 4 père .

Fin

Algorithme 4.2 : Etape d'opérateur croissement mutation.

4.5 Test d'arrêt :

Le critère d'arrêt indique que la solution est suffisamment proche de l'optimum. Plusieurs critères d'arrêt de l'algorithme sont possibles. On peut arrêter l'algorithme après un nombre suffisant de générations pour que l'espace de recherche soit convenablement exploré. On arrête l'algorithme lorsqu'on estime que la population n'évolue pas assez rapidement. On peut supposer qu'on est suffisamment proche de l'optimum. Il est à noter que dans tous les cas, aucune certitude concernant la bonne convergence de l'algorithme n'est assurée. La solution obtenue « en un temps fini » ne constitue qu'une approximation de l'optimum.
Notons que ce critère peut s'avérer couteux en temps de calcul si le nombre d'individus à traiter dans chaque population est important. C'est pourquoi, nous avons jugé suffisant d'approximer la solution après dix générations.

```
best wait time (16.3, (12, 5, 18, 5))

=== Gen 9 best solutions ===
best solution is [(16.3, (12.015540674851456, 5, 18.102672045065262, 5)),
(16.3, (12.030249845184494, 5, 18.46580016239733, 5)),
(16.3, (12.049764562119506, 5, 18.29862155381691, 5)),
 (16.3, (12.118795742496612, 5, 18.41973377377306, 5))]
```

Figure 18: Dernier génération avec le meilleur temps attente.

5. Combinaison entre simulation et optimisation :

Le problème d'optimisation étudié ainsi que sa méthode de résolution imposent le recours à la simulation. En effet, cette technique offre un prototype du monde réel qui évite les couts et risques d'un déploiement à échelle réel.
Nous avons choisi la simulation via SUMO pour deux principales raisons : c'est un outil gratuit et à utilisation académique à grande échelle [25].
La combinaison entre ces deux techniques, i.e. optimisation et simulation est réalisée conformément au schéma suivant :

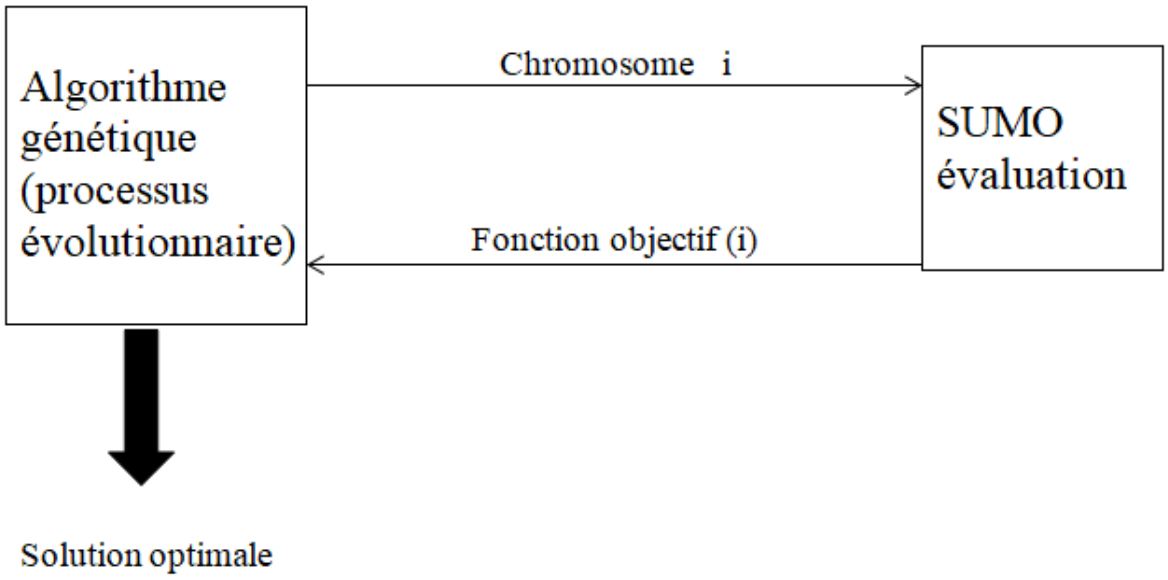

Figure 16 : Interaction entre simulation via SUMO et l'optimisation via algorithme génétique

L'organigramme ci-après détails encore plus aussi bien l'algorithme génétique que la mise en place la technique d'optimisation-simulation.

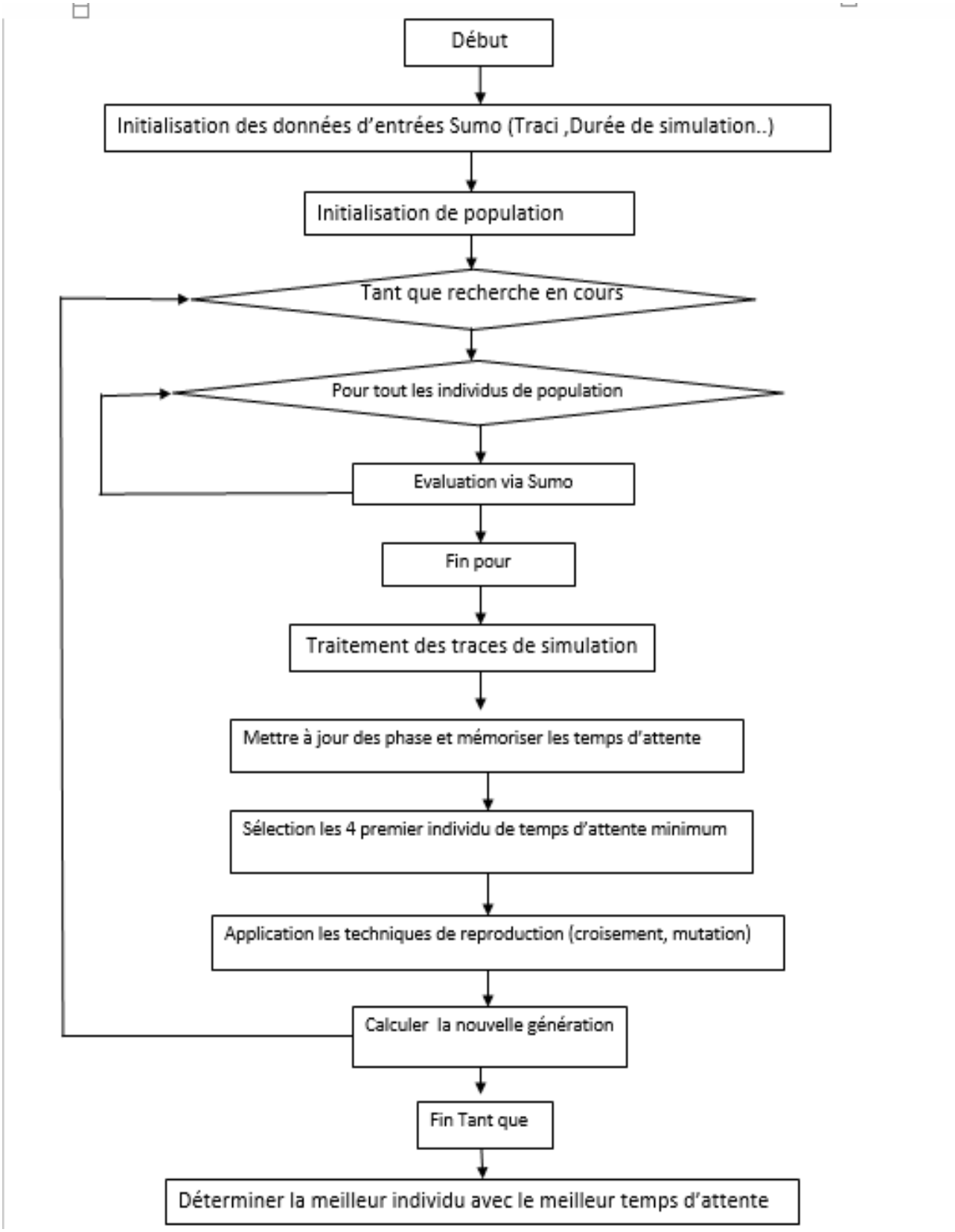

Figure 20 : Communication entre l'algorithme génétique et le simulateur SUMO.

6. Conclusion :

Dans ce chapitre,Nous avons expliqués les détails et la structure de notre travail.

Nous avons exposé en détail les différentes étapes qui constituent la structure générale du processus développé : codage, création de la population initiale, opération de croisement et de mutation

1. Introduction :

Pour valider les résultats de l'algorithme génétique, obtenus à partir de l'optimisation du problème de synchronisation des feux de circulation dans cette recherche, ces résultats a été comparés aux résultats de méthodes courantes et traditionnelles analytiquement utilisant la simulation sumo. Ceux-ci ont été utilisés pour déterminer le moment optimal pour chaque feu de circulation, dans le réseau et afin de réduire le temps d'attente pour chaque véhicule.

2. Démarche de test :

Pour tester les performances de l'algorithme génétique, en vue de réguler le trafic routier et optimiser les temps d'attente des véhicules, nous avons déterminé le réseau routier avec le flux des véhicules sur ce réseau. Nous avons choisi de travailler sur une intersection à Mila, à partir la quantité de flux qui l'emprunte au quotidien. La figure suivante montre une capture en SUMO-GUI du mouvement des véhicules sur l'intersection choisit. ainsi que des notifications sous forme de messages relatives aux nombre de véhicules en cours de déplacement sur le réseau et le temps de simulation.

Figure 21 : Mouvement des véhicules sur l'intersection

Le déroulement de nos simulations est contrôlé via TRACI pour mettre à jour les phases selon les meilleurs individus de l'algorithme génétique et ce à chaque génération.

Nous effectuons une étude comparative entre le contrôle via l'algorithme génétique et via feux fixe

Le contrôle via feux fixer est implémenté sans exécution de code python ; l'outil Netconvert de SUMO détermine le programme cyclique pour un seul feux et les enregistre sur le fichier.net.

Le contrôle via l'Algorithme Génétique est programmé sous python en interaction à SUMO. On a généré 10 génération, A chaque nouvelle génération on a simulé 10 individus de 4 gène (e1,e2,e3,e4). Chacun nous donnant 10 temps d'attente associés à ses propres individus. Nous avons choisi de chaque génération la plus petite valeur du temps d'attente (MinWatingTime) et leurs individus. Chaque individu il soumises à nouvelle simulation pour extraire les traces et analyser les performances de chacune. Ces calculées sur une période d'environ 1000s.

Après plusieurs exécutions et testes d'exécution des programmes avec des différentes individus. On a obtenu les résultats suivants de chaque exécution :

	e1	e2	e3	e4	temps d'attente
Exe1	10	5	17	5	15.62
Exe2	25	5	15	5	15.06
Exe3	20	5	26	5	16.2
Exe4	17	5	15	5	17.92
sumo	60	5	60	5	23.68

Tableau 12 : Résultats De L'Algorithme Génétique avec Temps d'attente

La dernière ligne représente l'exécution de SUMO pour des feux fixé sans contrôle supplémentaire en Algorithme Génétique, c.à.d. sans besoin des valeurs des individus.

On a fixé les Phase suivants : e1=25 ,e2=5 ,e3=12 ,e4=5 de meilleur temps d'attente .

Les traces des simulations de 10 générations sous forme de fichiers XML. Nous les exploitant pour créer des graphes comparatifs.

```
tripinfos xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://
    <tripinfo id="veh8" depart="7.00" departLane="-386231563#15_0" departPos="5.10" departSpeed="0.00"
    <tripinfo id="veh16" depart="13.00" departLane="171152952#6_0" departPos="5.10" departSpeed="0.00"
    <tripinfo id="veh71" depart="58.00" departLane="-729799830#0_0" departPos="5.10" departSpeed="0.00"
    <tripinfo id="veh24" depart="20.00" departLane="-693481637#0_0" departPos="5.10" departSpeed="0.00"
    <tripinfo id="veh25" depart="21.00" departLane="-386236569#0_0" departPos="5.10" departSpeed="0.00"
```

Figure 22 : Exemple de fichier tripinfo.xml

Id	Veh8
Depart	7.00
DepartLane	-386231563#15_0
DepartPos	5.10
DepartSpeed	0.00
DepartDelay	0.55
Arrival	84.00
ArrivalLane	171153024#3_0
ArrivalPos	60.14
ArrivalSpeed	12.99
Duration	77.00
RouteLength	779.74
WaitingTime	3.00
WaitingCount	1
StopTime	0.00
timeLoss	20.90
RerouteNo	1
Devices	tripinfo_veh8 routing_veh8
vType	veh_passenger
SpeedFactor	0.94
Vaporized	

Tableau 13: Les informations de fichier TripInfo de véhicule 8

```
summary xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="h
    <step time="0.00" loaded="2" inserted="1" running="1" waiting="0" ended="0" arrived="0" coll
    <step time="1.00" loaded="250" inserted="2" running="2" waiting="0" ended="0" arrived="0" co
    <step time="2.00" loaded="250" inserted="3" running="3" waiting="0" ended="0" arrived="0" co
    <step time="3.00" loaded="250" inserted="4" running="4" waiting="0" ended="0" arrived="0" co
    <step time="4.00" loaded="250" inserted="5" running="5" waiting="0" ended="0" arrived="0" co
    <step time="5.00" loaded="250" inserted="7" running="7" waiting="0" ended="0" arrived="0" co
    <step time="6.00" loaded="250" inserted="8" running="8" waiting="0" ended="0" arrived="0" co
    <step time="7.00" loaded="250" inserted="9" running="9" waiting="0" ended="0" arrived="0" co
    <step time="8.00" loaded="250" inserted="10" running="10" waiting="0" ended="0" arrived="0"
    <step time="9.00" loaded="250" inserted="12" running="12" waiting="0" ended="0" arrived="0"
```

Figure 17 : Exemple de fichier summary.xml

time	**0.00**
loaded	2
inserted	1
running	1
waiting	0
ended	0
arrived	0
collisions	0
teleports	0
halting	1
stopped	0
meanWaitingTime	0.00
meanTravelTime	-1.00
meanSpeed	0.00
meanSpeedRelative	0.00
duration	23927407

Tableau 14: Les informations de fichier Summary en step 0

3. Analyse Des Résultats :

Dans cette partie. Nous avons fait une comparaison entre les déférent exécution de programme, après l'analyse des résultat et la performance de chacune d'exécution: Exe1 , Exe2 ,Exe3 , Exe4 et Sumo .

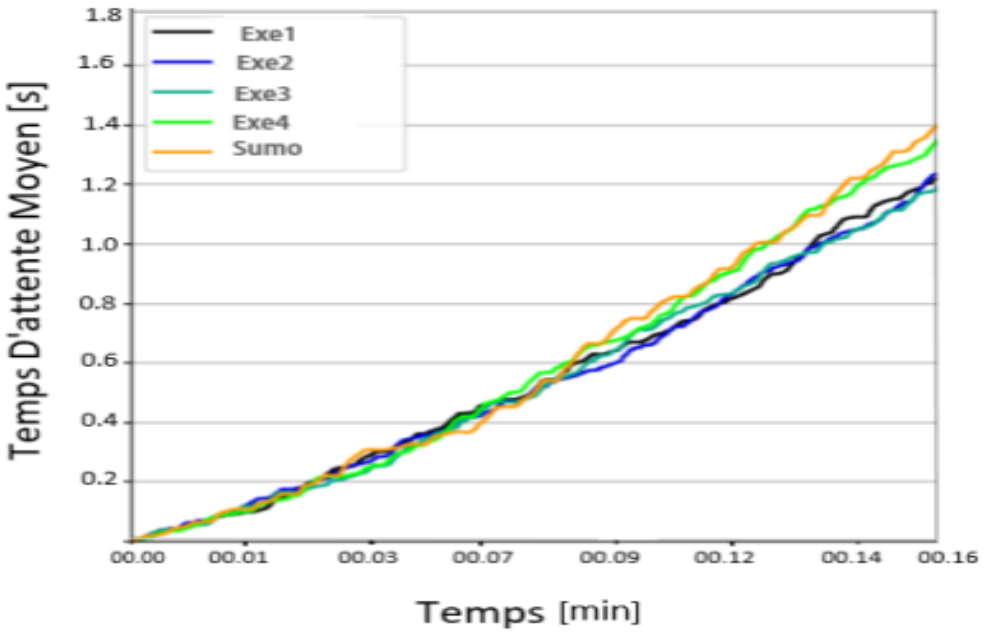

Figure 18 : Résultat de temps d'attente au fil de simulation

Cette figure montre le temps d'attente moyen de tous les véhicules sur l'intersection au fil du temps de simulation. On remarque que les résultats sont quasiment à égalité jusqu'à la 9eme minute, après cela, ils quoi ils commencent à varier, o l'on remarque une bonne amélioration des résultats de l'Algorithme Génétique par rapport à la constante. Ceci apparait plus clairement même lorsque le nombre de véhicules est sur route.

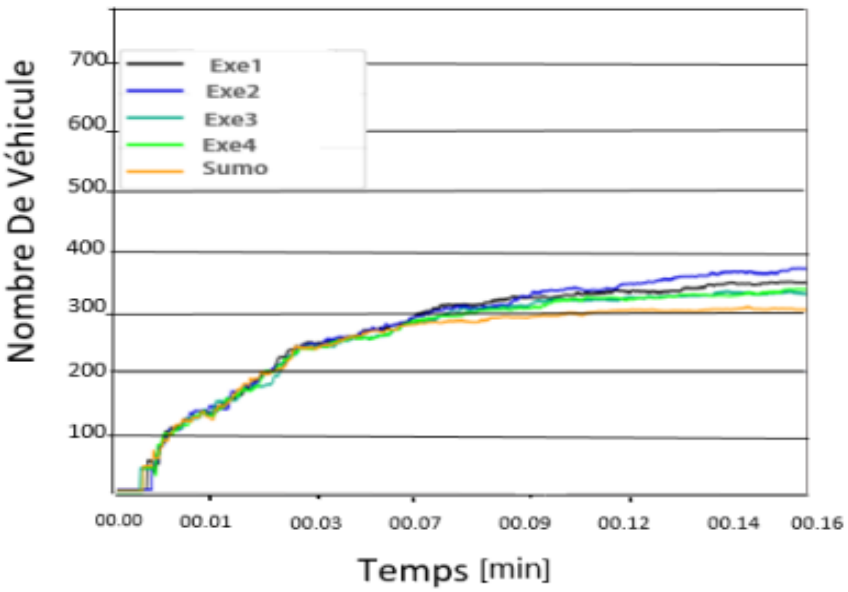

Figure 25 : Nombre de véhicule pendant le temps de simulation

Ce graphique montre le nombre de véhicule à chaque instant, durant le temps du simulation 16 min. Ont remarqué que l'algorithme génétique amélioré ce nombre de véhicules, par rapport l'exécution du sumo (300 véhicules). Et le résultat d'exécution du meilleur temps d'attente représente une bonne proportion dans le graphe (400 véhicules) atteignent leurs destinations.

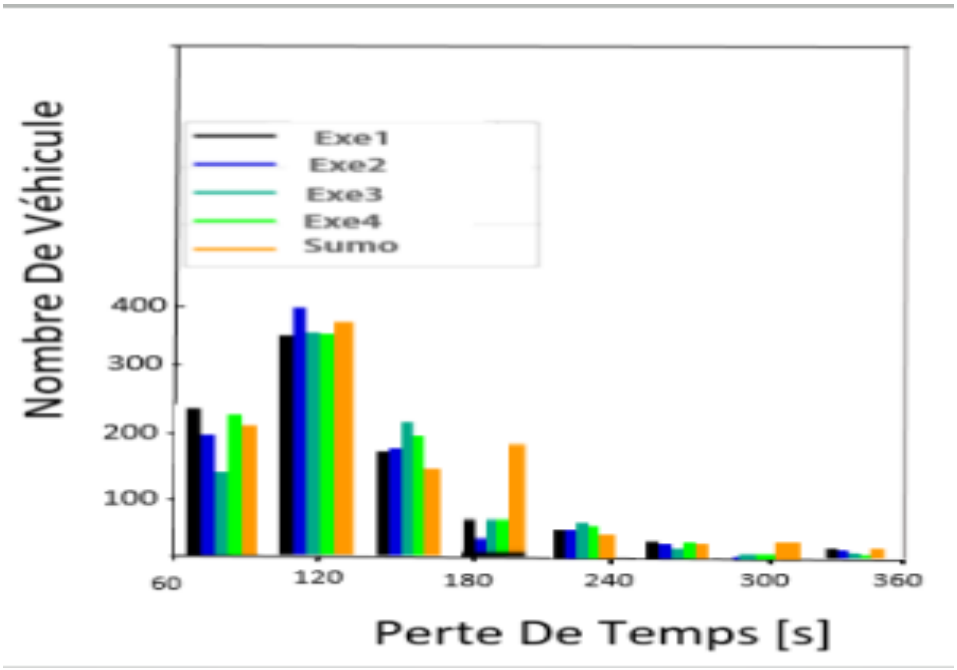

Figure 26 : Résultat de temps perdu

Le temps perdu (relevé par SUMO) est la métrique qui mesure pour chaque véhicule combien de temps il est contraint à conduire à une vitesse inférieure à l'idéale. Le graphe ci-haut montre les valeurs de temps perdu extraites de fichier Tripinfo.xml pour chaque période. La perte de temps est limitée, surtout dans les périodes de petites valeurs où le nombre de genoux est important. Généralement, le temps perdu pour compiler compensé par le temps gagné lors des simulations

4. Conclusion :

Dans ce chapitre, nous avons analysé les résultats des simulations et avons conclu que la simulation via l'algorithme Génétique, donne les meilleurs résultats de chaque exécution.

Conclusion générale

Les travaux et études sur les algorithmes génétiques sont unanimes : les algorithmes génétiques sont une méta-heuristique plus adaptée pour résoudre le problème de la synchronisation des feux de signalisation, que les méthodes traditionnelles. Ils donnent des résultats plus intéressants, à partir de peu de modifications.

Au début de notre travail, l'étape de documentation nous a permis de comprendre le principe de la technique de l'optimisation-simulation, ainsi que les notions élémentaires de l'algorithme génétique. À partir de ces connaissances, on a proposé d'évaluer la fonction objective qui minimise le temps d'attente durant plusieurs générations, pour obtenir la solution optimale. On est passé inévitablement par la simulation sur une zone de la ville de MILA. On a choisi le simulateur SUMO, pour implémenter la fonction objective et contrôler les feux de signalisation et réguler le trafic routier.

Le test de l'algorithme génétique sur notre réseau a abouti à diverses traces et diagrammes via SUMO. Les résultats analysés sont aussi favorables à l'utilisation de cet algorithme pour déduire les meilleurs programmes des feux et montrent l'apport de l'algorithme génétique pour optimiser le temps d'attente qui constitue notre fonction objective.

Comme perspective de notre travail, nous proposons de produire une interface graphique afin de faciliter l'accès aux ingénieurs de transport et de visualiser les résultats d'autres algorithmes. En ce sens, nous proposons également d'utiliser d'autres algorithmes métaheuristiques tel que PSO. De plus, un contrôle en temps réel des feux de signalisations sur de plus grandes étendues ou encore est une solution des plus avancées.

Références bibliographiques

[1] Güney GORGUN, Ibrahim Halil GUZELBEY: Simulation of traffic lights for green wave and dynamic change of signal, American Journal of Software Engineering and Applications, Vol. 2,

[2] ZINSALO Joël M. 2012 – 2013). "Techniques d'optimisation" .école polytechnique d'Abomey calavi (epac)- universite d'abomey calavi

[3] Luc Pellecuer . Sandrine Poteau. "La simulation" -cours de théorie de la circulation- civ 6705.

[4] ASSAM Amar.MOKDAD Mohamed dhia eddin. Mémoire Master Recherche : 'Une approche multi-agents pour la simulation de la mobilité urbaine et de la communication V2X' . UNIVERSITE AKLI MOAND OULHADJE-BOUIRA. 2018/2019

[5] Bastien Poggi. Submitted on 4 May 2015. Développement de concepts et outils d'aide à la décision pour l'optimisation via simulation. https: //hal.archives-ouvertes.fr/tel-01148522/document.

[6] Anne Chambard – ESI Group .Guillaume Colin de Verdière - CEA . Kambiz Kayvantash – CADLM. Philippe Mils – Thales. Philippe Ravier – Sylkan. Esther Slamitz – Systematic-Paris-Region. "Simulation Numérique Et Optimisation" -Feuille De Route 2020-

[7] Fei Yan. Contribution à la modélisation et à la régulation du trafic aux intersections : intégration des communications Vehicule-Infrastructure. Ordinateur et société [cs.CY]. Université de Technologie de Belfort-Montbeliard, 2012

[8] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning, Addison wesley.

[9] Davis, L. (1991). Handbook of Genetic Algorithms, Van Nostrand Reinhold.

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

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

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

[13] 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 – Setif.

[14] Souquet Amédée .Radet Francois-G. TE de fin d'année :'ALGORITHMES GENETIQUES'.Tutorat de Mr Philippe Audebaud. 1/06/2004

[15]Naima ZERARI. Mémoire Doctorat Recherche**:**' LES ALGORITHMES GENETIQUES EN MAINTENANCE'. UNIVERSITE EL HADJ LAKHDAR BATNA.Faculté des Sciences de l'Ingénieur.2006

[16]https: //sumo.dlr.de/docs/FAQ.html 08/08/2022.

[17] https: //sumo.dlr.de/docs/SUMO_at_a_Glance.html 10/08/2022.

[18] sumo - SUMO Documentation (dlr.de) 17/08/2022.

[19] Gudwin. R-R. Urban Traffic Simulation with SUMO. A Roadmap for the Beginners. 2016.
[20] TraCI - SUMO Documentation (dlr.de) 10/08/2022.
[21] https: //docplayer.fr/83780919-Introduction-au-logiciel-de-micro-simulation-de-la-circulation-vissim.html 10/08/2022.
[22] https: //fr.wikipedia.org/wiki/AnyLogic 12/08/2022.
[23] https: //www.anylogic.fr/road-traffic/ 13/08/2022.
[24] https: //www.anylogic.fr/features/ 13/08/2022.
[25] http : //sumo.dlr.de/wiki/M ainP age 17/08/2022.
[26] MEHIDID Fadila, Mémoire de fin d'étude ,Spécialité : Modélisation Contrôle et Optimisation thème Algorithme génétique , Soutenu le 19/06/2013.