Deanship of Graduate Studies
Al-Quds University

Benchmark for Tuning Metaheuristic Optimization Technique to Optimize Traffic Light Signals Timing

Rami Kamal Izzat Abu Shehab

M.Sc. Thesis

Jerusalem, Palestine

1437 / 2015

Benchmark for Tuning Metaheuristic Optimization Technique to Optimize Traffic Light Signals Timing

Prepared By:
Rami Kamal Izzat Abu Shehab

B.Sc.: Computer Science, Al-Quds University, Jerusalem Palestine

Supervisors: Dr. Badie Sartawi
Co-Superviser: Dr. Baker Abdalhaq

This thesis is submitted in partial fulfillment of requirements for the degree of Master of Computer Science / Department of Computer Science,
Faculty of Graduate Studies, Al-Quds University.

1437 / 2015

Al-Quds University
Deanship of Graduate Studies
Computer Science Department

Thesis Approval

Benchmark for Tuning Metaheuristic Optimization Technique to Optimize Traffic Light Signals Timing

Prepared By: Rami Kamal Izzat Abu Shehab
Registration No.: 20912301

Supervisors: Dr. Badie Sartawi
Co-Superviser: Dr. Baker Abdalhaq

Master thesis was submitted and accepted. Date: 10 / 12 /2015
The names and signatures of the examining committee members are as follows:

1- Head of Committee: Dr. Badie Sartawi. Signature: ………………..…..
2- Co-supervisor: Dr. Baker Abdalhaq. Signature: ………………..…..
3- Internal Examiner: Dr. Rasheed Jayousi. Signature: ………..………..
4- External Examiner: Dr. Emad Barhoumi. Signature: ……..…………..

Jerusalem, Palestine

1437 / 2015

Dedication

This work is dedicated…

To my parents, for their love, endless support and encouragement…

To my beloved wife: without her caring support, it would not have been possible…

To my son and daughter …. Yazan, Yamamah, Jalal

To my brothers, sisters, friends and colleagues…

To all of you I say a big
“Thank you” for being example of love and care.

Rami K. I. Abu Shehab

Declaration

I certify that this thesis, submitted for the degree of Master, is the result of my own research, except where otherwise acknowledged, and this study (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed…………………………

Rami K. I. Abu Shehab

Date: 10 / 12 /2015

Acknowledgments

First and foremost Praise be to the Almighty Allah, Lord of all creatures, the Most Gracious and Most Merciful, for all the blessings I have experienced throughout my life.

My sincere thanks are to my supervisors, Dr. Badie Sartawi and Dr. Baker Abdalhaq, for their sincere effort, interest and time they have given me to do this research.

Thanks are also to the Examining Committee Members.

I am very grateful to all professors in the Computer Science Department of Al-Quds University for their support and dedication.

I am also extremely grateful and indebted to the dean of College of Engineering, Dr. Khaled Alsahli, and to An-Najah National University for giving me permission to use its computer laboratory.

I thank my mother for her generous love and support, my father for encouragement and motivation, and my family members for their support.

Finally, I would like to thank my wife. Her support, encouragement, quiet patience and unwavering love were undeniable.

Abstract

Traffic congestion at intersections is an international problem in the cities. This problem causes more waiting time, air pollution, petrol consumption, stress of people and healthy problems. Against this background, this research presents a benchmark iterative approach for optimal use of the metaheuristic optimization techniques to optimize the traffic light signals timing problem. A good control of the traffic light signals timing on road networks may help in solving the traffic congestion problems. The aim of this research is to identify the most suitable metaheuristic optimization technique to optimize the traffic light signals timing problem, thus reducing average travel time (*ATT*) for each vehicle, waiting time, petrol consumption by vehicles and air pollution to the lowest possible level/degree.

The central part of Nablus road network has a huge traffic congestion at the traffic light signals. It was selected as a research case study and was represented by the SUMO simulator. The researcher used a random algorithm and three different metaheuristic optimization techniques: three types of Genetic Algorithm (GA), Particle Swarm Algorithm (PS) and five types of Tabu Search Algorithm (TS). Parameters in each metaheuristic algorithm affect the efficiency of the algorithm in finding the optimal solutions. The best values of these parameters are difficult to be determined; their values were assumed in the previous traffic light signals timing optimization research. The efficiency of the metaheuristic algorithm cannot be ascertained of being good or bad. Therefore, the values of these parameters need a tuning process but this cannot be done by using SUMO simulator because of its heavy computation.

The researcher used a benchmark iterative approach to tune the values of the metaheuristic algorithm parameters by using a benchmark function. The chosen function has

similar characteristics to the traffic light signals timing problem. Then, through the use of this approach, the researcher arrived at the optimal use of the metaheuristic optimization algorithms to optimize traffic light signals timing problem. The efficiency of each metaheuristic optimization algorithm, tested in this research, is in finding the optimal or near optimal solution after using the benchmark iterative approach. The results of metaheuristic optimization algorithm improved at some values of the tuned parameters.

The researcher validated the research results by comparing average results of the metaheuristic algorithms, used in solving the traffic light signals optimization problem after using benchmark iterative approach, with the average results of the same metaheuristic algorithms used before using the benchmark iterative approach; they were also compared with the results of Webster, HCM methods and SYNCHRO simulator.

In the light of these study findings, the researcher recommends trying the benchmark iterative approach to get more efficient solutions which are very close to the optimal solution for the traffic light signals timing optimization problem and many complex practical optimization problems that we face in real life.

ضبط خوارزميات التحسين التخمينية لتحسين توقيت اشارات المرور الضوئية بقارنة الدوال.

إعداد : رامي كمال عزت ابو شهاب.

اشراف: د. بديع السرطاوي و د. بكر عبدالحق.

ملخص:

الازدحامات المرورية عند التقاطعات هي مشكله عالمية في المدن. هذه المشكلة تسبب المزيد من وقت الانتظار وتلوث الهواء واستهلاك الوقود، و توتر الناس ومشاكل صحية. على هذه الخلفية، يقدم هذا البحث نهج المعيار المكرر للاستخدام تقنيات التحسين التخمينية في تحسين مشكلة توقيت الإشارات الضوئية. التحكم الجيد في توقيت الاشارات الضوئية على شبكات الطرق قد يساعد في حل مشاكل الازدحام المروري. يهدف هذا البحث الى تحديد أفضل و أنسب تقنية تحسين تخمينية لتحسين مشكلة توقيت الاشارات الضوئية، وبالتالي تقليل متوسط الوقت الذي يستغرقه السفر (ATT)لكل مركبة، و وقت الانتظار، و استهلاك الوقود المستخدم في المركبات وتلوث الهواء إلى أدنى مستوى ممكن.

يعاني الجزء المركزي من شبكة طرق مدينة نابلس من ازدحام مروري كبير على الاشارات الضوئية. و تم اختيار هذا الجزء كحالة البحث الدراسية و التي تم تمثيلها باستخدام برنامج المحاكاة سومو. و استخدم الباحث خوارزمية عشوائية و ثلاث تقنيات تحسين تخمينية و هي: ثلاث انواع من الخوارزمية الجينية، و خورزمية سرب الجسيمات، و خمسة انواع من خوارزمية التابو. و هناك متغيرات في كل خوارزمية **تخمينية** تؤثر على فعالية الخوارمية في ايجاد الحلول المثلى. و من الصعب تحديد افضل القيم لهذه المتغيرات؛ و قيم هذه المتغيرات كانت تفترض في ابحات تحسين توقيت الاشارات الضوئية السابقة. وفي هذه الحاله فعالية اقتران التحسين **التخميني** لا يمكن التحقق منها اذا ما كانت جيده او سيئة. ولذلك فان قيم هذه المتغيرات بحاجه لعملية ضبط ، ولكن لا يمكننا ذلك باستخدام برنامج المحاكاه سومو لانه حساباته ثقيله و طويله.

استخدم الباحث طريقة مقارنة الدوال لضبط قيم متغيرات خوارزمية التحسين التخمينية باستخدام خوارزمية معيار. خوارمية المعيار المختاره لها خصائص شبيهه بمشكلة توقيت الاشارات الضوئية. ثم من خلال استخدام هذه الطريقة، وصل الباحث الى افضل استخدام لخوارزميات التحسين التخمينية لتحسين مشكلة توقيت الاشارات الاضوئية. وفي هذا البحث تم اختبار فعالية كل خوارمية تحسين تخمينية في ايجاد الحل الامثل او حل قريب من الحل الامثل بعد ضبط خوارزمية التحسين التخمينية. لقد تحسنت نتائج خوارزمية التحسين التخمينية عند بعض قيم المتغيرات التي تم ضبطها.

قام الباحث بالتحقق من نتائج البحث بمقارنة معدل نتائج خوارزميات التحسين التخمينية التي امستخدمها في تحسين مشكلة توقيت الاشارات الضوئية قبل ضبط خوارزمية التحسين التخمينية، مع معدل نتائج نفس الخوارزميات التخمينية التي امستخدمها بعد ضبط خوارزمية التحسين التخمينية؛ وهذه النتائج تمت مقارنتها مع نتائج طريقتي ويبستر و HCM و برنامج السنكرو.

في ضوء نتائج هذه الدراسة، يوصي الباحث بتجريب طريقة مقارنة الدوال لضبط خوارزميات التحسين التخمينية للحصول على حلول فعالة اكثر و التي تكون قريبة جدا من الحل الامثل لتحسين مشكلة توقيت الاشارات الضوئية و لتحسين المشاكل العملية المعقدة التي تواجهنا في الحياة العملية.

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter One

Introduction

Metropolitan cities in many countries suffer from chronic traffic congestion problems on their road networks. The major traffic congestion is mostly at the intersections and traffic light signals. These congestion problems cause many problems such as waste of time, waste of petrol, and air pollution, stress and other health problems for people.

In 2002, people spent 100,000 hours in total in queues in the Greater Copenhagen road infrastructure, and nearly 750 million Euros were lost, due to traffic problems [1]. Danish researchers and Copenhagen municipality tried to solve the traffic congestion problem, through improvement of road network, by increasing number of lanes, and building of more bridges. But the problem is that it is very difficult to increase number of lanes and bridges in all cities because of lack of open spaces within urban centers and lack of financial resources to build bridges. In the light of this challenge, there is a chance for alleviation of this problem through improvement of the control process of the traffic light signals timing. If the optimal time for each traffic light signal was selected, the traffic congestion problem might be reduced [2, 3, 4, 5, 6, 7]. However, the selection process for the optimal traffic light signals timing is a very complex problem because the road networks have a random behavior [8].

Engineers and scientists have developed simulation tools to represent the road network and to compute very huge parameter values such positions, speeds, accelerates, decelerates …, for all vehicles, and for control of the road network, to help in solving the road network problems by using the force of computer systems. One suitable way to solve the traffic light signals timing problem is by using optimization processes. One suitable optimization technique, to solve this type of problems, is using the metaheuristic optimization technique as the literature review shows below in the next chapter.

The case study of this research was a Nablus city center road network which was represented on SUMO simulation (Simulation of Urban MObility). The case study contained 13 traffic light signals. Each traffic light signal may have a red or green light and takes a fixed time value between 10s to 100s, and each traffic light signal had 90 probabilities. All parameters of the road network of the case study for the network road in SUMO simulation were stable, and the variable inputs were just a list of 13 traffic light signals timing. To find the optimal traffic light signals timing, which has a minimum average travel time (*ATT*) for each vehicle, a search process in a large solution space had to be done.

The execution time of SUMO simulation for one hour, for each input time list, takes the computer a ½ minute to complete. Many years would be needed to try all different probabilities of input time lists. This is a strong proof of the complexity of the problem. Because this problem is random and unsolvable by traditional analytical mathematic [2, 9, 10], the researchers used different metaheuristic optimization techniques, as previous research shows, to optimize the traffic light signals timing in the road network of the case study [2, 3, 4, 5, 6, 7, 9, 11].

The researcher used a basis random algorithm and nine metaheuristic optimization methods to optimize the traffic light signals timing problem in the case study. These methods were three types of Genetic Algorithm (GA), Particle Swarm algorithm (PS) and five types of Tabu Search algorithm (TS). One of them was the main algorithm and four types were developed from the main TS algorithm type.

The researcher selected GA type1, PS and TS type1 algorithms from Abdalhaq B. and Abu Baker M. research in which they had calibrated many parameters of SUMO simulators like vehicle length, minimum and maximum speed, acceleration, deceleration. These algorithms were used in addition to Simultaneous Perturbation for Stochastic Approximation algorithm (SPSA). The results of these metaheuristic algorithms were compared with the classical optimization algorithms like Nealder-Mead (NM) and Constrained Optimization By Linear Approximation algorithm (COBYLA) [12].

However, each metaheuristic algorithm had main parameters whose values had an effect on the efficiency (ex. arriving speed to the optimal or near optimal solution) of the algorithm in finding the optimal solution. The values of these parameters were assumed in the previous traffic light signals timing optimization researches, so that we cannot determine if algorithm efficiency is good or not (see next chapter). These parameters are as follows:

1- Iteration number: In a lot of research, the number of iterations was executed to find the whether the optimal solution was presupposed, but these iterations failed to determine whether the best solution obtained was near or far from the optimal solution. A number of iterations were used in PS and TS algorithms. The number of iterations was equal to the generation size multiplied by the generation numbers in GA.
2- *W*, *cp*, *cg* parameters: These parameters were used in PS algorithm, and their values had an effect on the behavior of this algorithm [12, 13], because *w* represents the percentage effect of the particle's speed and vector itself, while *cp* represents the percentage effect of the best neighborhood particle's speed and vector, and *cg* parameter represents the percentage effect of the best particle's speed and vector in the swarm.

3- *k* and *tau* numbers: These two parameters were used in TS algorithm. *k* represents the neighborhood steps from the best solution in the generation by adding to and dropping *k* from each record to get a new generation. *tau* represents a value for the best solution arrived at to avoid the searching process in the same region for *tau* values, and to reroute the searching process to a new region for *tau* iterations[12].

4- GA mutation rate, crossover rate and selection rate parameters

To determine the best parameters' values, all values' probabilities had to be tried on the traffic light signals timing problem of the case study with many replications. However, if SUMO simulator execution time for one solution by one computer was ½ minute, then this process would need a very long time for each metaheuristic algorithm, because the number of probabilities is large and the number of replications has to be large to get a confident solution.

Research Problem

How can we determine the best parameters' values of each metaheuristic optimization algorithm for optimizing the traffic light signals timing problem?

Research questions

Can we find a simple approach to determine the best previous parameters' values in a short time for efficient metaheuristic optimization algorithms to optimize the traffic light signals timing problem?

Can we improve some metaheuristic optimization algorithms to find new solutions that are close to the optimal solution in solving the traffic light signals timing optimization problem?

Can we determine the best metaheuristic optimization algorithm suitable for solving traffic light signals timing optimization problem?

To answer these questions, the researcher used a benchmark iterative approach, for tuning the parameters values of the metaheuristic optimization algorithms, to optimize the practical problem in a more efficient way than those of the previous research. The methodology of this research depends on experimentation to determine the best parameters' values for any metaheuristic optimization method at a reasonable time by using a suitable light benchmark function.

The benchmark iterative approach is used to optimize the traffic light signals timing problem to help in answering Teklu F., Sumalee A., Watling D. (2007) question: How will the best parameters' values determine an efficient metaheuristic optimization algorithm to solve the traffic light signals timing optimization problem? [4].

Many experiments have been conducted, using metaheuristic algorithms to compute the average travel time for each vehicle at each algorithm with many replications. At the end, comparative processes, between the average results of the algorithms, were done to determine the efficiency of the metaheuristic algorithm when the benchmark iterative approach was used, and the most suitable algorithm type to solve traffic light signals timing problem was determined. These algorithms' results were validated by comparing the best algorithms' results for optimizing the traffic light signals timing problem before using benchmark iterative approach and after using benchmark iterative approach, and with two main mathematical models for traffic light timing Webster and HCM methods [14], and SYNCHRO simulation timing results [15]. At the end, it was found that the results were promising.

1.1 Motivation

This study seeks first to provide metaheuristic optimization researchers with a benchmark iterative approach for optimal use of metaheuristic optimization techniques to get more efficient solutions for their problems. Second, it endeavors to prove that it is possible to benefit from the metaheuristic optimization techniques in an optimal way to solve complex practical optimization problems. Third, it aims at applying computational methodologies to solve traffic engineering problems for civil and traffic engineering. Fourth, it also aims at reducing the problems of the traffic congestion, resulting from the control of the traffic light signals, to the lowest level.

1.2 Objectives

This study seeks to find an approach for optimal use of metaheuristic optimization techniques to obtain new solutions which are close to the optimal solution. It also seeks to find the most suitable metaheuristic optimization algorithm to optimize the practical problem such as the traffic light signals timing problem. Finally, this study aims at improving metaheuristic algorithms of practical problems to obtain new good solutions.

1.3 Research Contribution

1. Using a benchmark iterative approach for optimal use of the metaheuristic optimization techniques to solve complex practical optimization problems, and find the optimal or near optimal solution.
2. Improving the TS algorithm to get a global minimum for the benchmark function, and get good solutions for the traffic light signals timing optimization problem compared with the basic TS algorithms.

It is worth noting that as a result of this research, a scientific research paper was accepted and published in the *Proceedings of the 6th International Conference on Computer Science & Information Technology CSIT* (see IEEE Computer website [16].

This thesis is organized as follows: Review of literature will be discussed in chapter two. Research background and details, including a description of research problem, case study and data, tools, techniques and methods used in this research, are presented in chapter three. Chapter four is devoted to the methodology of the study and the experiments. Experimental results are presented and discussed in chapter five. Experimental results validity is presented in chapter six. Finally, Conclusion and recommendations are given in chapter seven.

Chapter Two

Review of Literature

This chapter summarizes related literature on metaheuristic optimization algorithms. More specifically, it sheds light on different traffic light signals timing optimization problem methodologies and tuning parameters' values for metaheuristic optimization algorithm. It also provides an introduction to optimization and traces the development and classification of different optimization techniques, and traffic road network models and simulations.

2. 1 Methodologies of Traffic Light Signals Timing Optimization Problem

Methodology is a set of steps, tools and methods which can be used in problem solving processes. A lot of research has been conducted on the optimization process for the traffic light signals timing problem. Webster (1958) was the first researcher to introduce/ define a new methodology for solving this problem [14]. His methodology, which depends on the first development of mathematical model, aimed at optimizing the cycle time length and green time for the traffic light signals at an isolated intersection to minimize the delay time, queue length, or the volume of vehicles. The main shortcoming of this model is that it was used to optimize just one isolated intersection.

Researches in this field, after that, have developed approaches, models, algorithms for improving the solutions of the traffic congestions on the intersections depending on Webster's models. The researchers suggested that using metaheuristic optimization techniques might be more efficient to optimize many intersections as a global road network traffic light signals timing in comparison with Webster's method used to synchronize the movements of vehicles through the traffic light signals. A number of researchers used metaheuristic optimization algorithms for optimizing the traffic light signals timing problem to reduce the traffic congestion problems.

Yun I. and Park B. (2006) used a methodology which depends on simulating a real road network in Virginia, USA. This methodology contained 82 traffic signals at 12 intersections. CORIM microscopic traffic simulation model [2], and heuristic optimization methods GA, Simulate of Annealing (SA) and Opt Quest Engine, with iteration numbers= 2,500, were used. They used these algorithms to optimize traffic light signals timing (cycle time) to minimize the total queue time such as delay time for each vehicle. Then they compared their results with the results of another existent network optimization program named SYNCHRO which is based on macroscopic models. Their conclusion was that GA results were more suitable than other methods' results. SYNCHRO program results were used.

Teklu F., Sumalee A. and Watling D. (2007) used a methodology to solve a signal timing problem based on NDP (Network Design Problem). They used GA with a different population size={30, 50, 70}, and a different generation number= {150, 90, 70} to optimize the traffic signal cycle time to minimize the total travel time to deal with congestion problems and atmospheric pollution [4]. The researchers developed an application called GA-FITSUM which depended on SATURN model; then they implemented their case study which was road network in the city of Chester, UK, which had 75 signalized junctions. The operations of the GA used were selection processes for the best chromosomes with higher probabilities named elitism. They were randomly paired crossover for parents to produce offspring, and the mutation process on genes, on some probabilistically selected chromosomes, was followed to mutate and form the next population. They compared GA-FITSUM results TT with SATOPT optimized signal timings. GA- FITSUM was found to have more promising results in finding the optimum signal timings than SATOPT. During their research, the researchers observed that there was a relationship between the GA parameters' values, like population size and generation number, and the performance of the algorithm. In this context, the researchers asked themselves one question: How can we find optimum values of the GA parameters? However, the answer is still controversial and requires further research.

Medina J., Moreno M., Cabrera M. and Royo E. (2009) proposed a methodology to optimize the traffic signal states sequence through simulation of a virtual road network on a microscopic simulator model, developed in this research as a non-linear model for simulating traffic behavior based on the Cellular Automata Model [11]. Then they used GA to fit the number of vehicles that left the network. They carried out a traffic simulation to speed up their work; they developed a cluster system based on a Beowulf Cluster. In this research, each traffic signal had two states, red light which means stop, and yellow light which means go, and all traffic light signals were encoded in one chromosome, and each gene was encoded as red='1', and yellow='0'. In their experiment, they presupposed that the population size=200 chromosome, generation number=200, and the first generation would be created randomly. Then the next generation would be created by selection of the best two chromosomes; later they conducted a crossover between parents, a mutation of the parents and new chromosomes with some probabilities. The conclusion was that this methodology could be helpful to solve the traffic congestions on the road network.

Singh D. and Singh R. (2009) used a methodology which depends on development of a mathematical model derived from Webster's and HCM's models. They computed and estimated the delay time at isolated intersections by some equations, and used GA, with a fixed population size=20 and generation number=50, to optimize traffic signals timing plan to minimize delay time parameter. Then they implemented their real network on a MATLAB program as a simulation tool, and at the end, they compared the results of the estimation equations model [7]. The results of the optimization process through GA, and the delay time was reduced by using GA more than estimation equations.

Farooqi A., Munir A. and Baig R. (2009)'s methodology depends on a newly developed simulation tool named THE simulator. They used it to simulate their virtual traffic road network which had 16 traffic signals [3]. Then they used GA, with population size=10 and generation number=10, to optimized the traffic light signals time cycle to minimize the delay time and the travel time. The results were found to be good because the total wait time was reduced.

Sklenar J., Beranek Z. and Popela P. (2009)'s methodology depends on developing a new traffic simulator named Stochastic Simulation in Java library for stochastic simulation. A numerical approximation objective function was developed, but the differential algorithms failed to optimize this function [8]. The researchers implemented a heuristic optimization algorithm SA to optimize the traffic light signals timing parameters of three junctions, at Konecného Square in Brno, The Czech Republic, to maximize the throughput and to minimize the average waiting time in the system. The distributions of the vehicles' arrival queue rate and the time of services for the vehicles at the traffic light signals were not exponential as they had supposed, so this problem could not be solved by using Jackson network and queuing theory as an analytical way. Using the simulation tool was found to be more efficient. At the end, the current timing plan was improved for the junctions, so the waiting time, and the queue length were reduced. The results were compared with the results of VISSIM model which was developed by BKOM.

In their methodology, Singh L., Tripathi S. and Arora H. (2009) depended on development of an emulator to represent dynamic conditions of traffic on an isolated intersection with traffic system conditions controlled with a surveillance system (by traffic camera). They wanted to provide a real time status to the optimization control algorithm which was decided for traffic light signals status. They used GA to optimize traffic light signals timing problem to determine the best time duration for the signal and to reduce stops and overall vehicle delays, or to maximize the throughput [5]. They sought to develop an efficient traffic adaptive control strategy that would identify a real time traffic scenario. The researchers compared the results that were obtained by presupposing real time based on traditional fixed time system. The results of real time based on the system were found to have more significant performance.

Renfrew D. (2009) investigated a new approach to find the optimal signal timing plan for a traffic intersection using ant colony optimization algorithms to optimize traffic signal timing on the intersection and minimize vehicle delays [6]. They considered an isolated traffic intersection, and used six types of the ACO algorithm; Ant System Algorithm, Ant System with local search algorithm, Elitist Ant System Algorithm, Elitist Ant System with local search algorithm, Elitist Ant System with local search and heuristics algorithm, and Rank-based Ant System with local search and heuristics, at these parameters' values, ants number={10, 25, 50}, iteration number=100, the pheromone evaporation coefficients of p=4 and p=2 and the vehicle arrival rate is 800 vph (vehicles per hour) per movement.

The convergence rate parameter of pheromone concentration for optimal solution, in different algorithm types of the ACO, was tested. The best simulation results were found in the Rank-Based Ant System algorithm. The average vehicle delay was tested and compared with a fully traditional actuated control.

Table 2.1: Traffic light signals timing optimization of different research.

Study	Methodology components and conclusion of each research					
	Optimization Methods	**Simulation Tool**	**Optimization Parameter**	**Fit Function**	**Network Type**	**Conclusion**
[2]	GA & SA & OptQuest Engine	CORIM microscopic Model	Signal timing	Minimize the total Queue time, and delay time	Fairfax, Virginia, USA road network	GA is the best
[4]	GA	SATURN model	Traffic signal cycle time	Minimize the total time	City of Chester in UK	Positive
[11]	GA	Microscopic model	Signal timing	Maximize # of vehicles lifted the network	Virtual Network	Positive
[7]	GA	MATLAB	delay time.	Minimize the total delay.	Virtual network	Positive
[3]	GA	THE simulator	Signal timing	Minimize delay time and total travel time	Virtual network	Positive
[8]	SA	Stochastic simulation model	Signal timing	Maximize network throughput and Minimize waiting time	Konecného square in Brno, Czech Republic	Positive
[5]	GA	Their own emulator	Signal timing	Maximize network throughput and Minimize total delay time	Isolated four-way intersection	Positive
[6]	ACO & fully actuated algorithm	Dynamic simulator traffic model	Signal timing	Minimize total delay time	Four-lagged isolated traffic intersection	ACO is the best

The results showed that when vehicles' flow was less than 600vph, the conventional fully actuated control was more efficient, but when the vehicles' flow greater than 600vph, ACO algorithm was more efficient.

Table 2.1 summarizes the traffic light signals timing optimization methods of previous research. And the differences between the previous traffic light signals timing optimization problem and this research can be summarized as follows:

1- The main shortcoming of previous research was in the researchers' methodologies. They just used some types of the metaheuristic optimization algorithm without any improvement of the algorithms used in traffic light signals timing problem or any tuning processes for the algorithms' parameters.

2- In this research, a benchmark iterative approach for using is used to tune many metaheuristic algorithms' parameters' values, by using a benchmark function to improve the algorithms' performance in a short time. The proposed simple approach is used to improve several metaheuristic algorithms such as GA, PS and TS.
3- In this research, through the use of the benchmark iterative approach, new types of TS algorithm were developed. The results were found to be better than basic TS algorithm in both optimizing benchmark function and traffic light signals timing optimization problems.

2.2 Parameters Tuning for Metaheuristic Optimization Algorithms

Metaheuristic optimization techniques began to be used recently at a large scale to solve complex problems, which are unsolvable by traditional optimization methods. Each one of these techniques has many parameters or factors which take huge value probabilities. These parameters demonstrate the algorithm's efficiency. Finding a process for the best parameters' values which the metaheuristic algorithm be more efficient in finding optimal or near optimal solutions named parameter tuning.

Universally optimal parameters' values for some algorithms to solve many problems don't exist yet. Finding the best parameters' values is not an easy task and it is very difficult to understand the effect of the algorithm's parameters on each problem [17]. The problem is that some algorithms work efficiently at the best parameters' values on some problems, but may not work efficiently on others. So this problem is a big challenge for each optimization problem [18]. Therefore, using a set of benchmark problems may help in the tuning process.

Xu J., Chiu S. and Glover F. (1998) proposed a systematic procedure to tune five parameters of TS algorithm: neighborhood structure and moves, move evaluation and error correction, TS memories, probabilistic move selection, and advanced restarting and recovery. In order to improve the algorithm performance to solve many test problems, they used statistical tests to ensure the best parameters' values. After conducting the tests, they found that the procedure of fine-tuning has improved the performance of TS algorithm for a telecommunications network design problem [19].

Balci H. and Valenzuela J. (2004) used a new approach to solve UCP (Unit Commitment Problem) problems. They depended on a method that uses a combination of PS algorithm and LR (Langrangian relaxation) framework; they tuned *w*, *cp*, *cg* and the number of iteration parameters for the new method to solve a six instance UCP problem. The results of the proposed method were found to be more computationally efficient than other methods which were used in this research. Therefore, tuning PS algorithm parameters, like *w*, *cp*, *cg* and number of iterations, are very important according to the efficiency of the algorithm to find optimal or near optimal solutions [20].

In their study, S. Smit and A. Eiben (2009) focused on parameter tuning to find out the advantage of tuning metaheuristic algorithms to improve the algorithm performance. They selected 10 benchmark test functions to determine the best values of population size, offspring size, mutation probability, crossover points, crossover probability and tournament size parameters for the GA. They concluded that the optimal selection for the parameters' values for these problems appeared to have more efficient results than common selection [18].

Pedersen M. (2010) tuned the parameters of PS and new type of PS called MOL (Many Optimizing Liaisons) algorithms' (*s (swarm size)*, *w*, *cp*, *cg*) and *s*, *w*, *cg* to solve 12 benchmark problems. The results of MOL algorithms were found to be more efficient in several benchmark problems than PS algorithm [21].

H. Akbaripour and E. Masehian (2013) presented a new framework for metaheuristic algorithms' parameters tuning. They claimed that parameters tuning for metaheuristic algorithm could improve the efficiency and capability of the functions to find the optimal or near optimal solutions. Their framework depended on determining parameter levels and ranges, using Designs of Experiment, Signal to Noise ration, Shannon entropy and VIKOR methods. Tuned SA algorithms were found to be more efficient in solving N-queen problem. The results of tuned GA were also found to be more efficient in solving Hub Location Problem [17].

From the findings of these studies, one infers how much important the parameters tuning is for the efficiency of metaheuristic optimization algorithms, and how we can tune the parameters. The main parameters for each metaheuristic optimization algorithms were determined. Benchmark functions are used in this research.

2.3 Introduction to Optimization

Optimum means the best. The optimization process is the process that determines the conditions which produce the best possible results, and the best results are when they have maximum or minimum values. The main goal of the optimization process is either to minimize effort or to maximize the benefits especially when it comes to taking decisions pertinent to design, construction, maintenance, …, and engineering. The effort and benefit can be usually presented as a function of certain variables, hence the optimization process is the process of finding the variables' values which obtain a maximum or a minimum result of a function [9], and "There is no single available method for solving all optimization problems efficiently. The optimum seeking methods are also known as mathematical programming techniques" [9], p.1.

Mathematical programming consists of many techniques such as calculus method, calculus of variations, nonlinear programming, geometric programming, quadratic programming, linear programming, dynamic programming, integer programming, stochastic programming, separable programming, multi objective programming, game theory, simulated annealing, genetic algorithms and neural network. In this context, programming doesn't mean software, but it means planning for a solution [9].

2.3.1. Historical Development

Optimization science could be traced back to the days of Newton, Cauchy and Lagrange [9]. Newton and Leibnitz's contributions to calculus were developed for the differential methods. Calculus of variations, which deals with the minimization of functions, was developed by Bernoulli, Euler, Lagrange and Weirstrass. In the middle of twentieth century, when computers' speed increased remarkably, the implementation of the procedures and stimulated research on new methods became possible.

To solve linear programming problems, simplex methods were developed by Dantzig in 1947. The optimal principle, developed in 1957 by Bellman to solve the dynamic programming problems, paved the way for constrained optimization methods development. During the early 1960s, unconstrained optimization methods, such as nonlinear programming, were developed significantly by Zoutendijk and Rosen. In the 1960s, Duffin, Zener and Peterson developed geometric programming. Integer programming, developed by Gomory, was one of the most common world applications under this category of problems. Then stochastic programming techniques were developed by Dantzing, Chares and Cooper. The goal of programming was developed to solve specific types of multi-objective optimization problems which were proposed for linear programming by Cooper in 1961. And Game's theory, developed by Neumann in 1928, was applied to solve several mathematical economic and military problems, but in recent years this theory has been applied to solve the design of engineering problems.

Metaheuristic optimization techniques, such as genetic algorithms, simulated annealing and neural network methods, are a new approach of mathematical programming techniques. Genetic algorithms are search techniques which are based on natural selection and genetics.

2.3.2. Optimization Methods for Optimization Problems

Optimization processes can be applied to solve any engineering problems such as design of aircrafts, bridges, towers, industries, machines, and control systems. The main mathematical statement for any optimization problem is called "the objective functions" [9] as below:

cost function: ***Min. f(x)***

or ***max.f(x)***

Constraints: $x \in R^n$

$$\text{Find } \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \end{pmatrix} \textbf{ which min. } f(x) \textbf{ or max. } f(x) \ldots\ldots\ldots\ldots\ldots (2.1)$$

Under constraint: $a <= x_i <= b$.

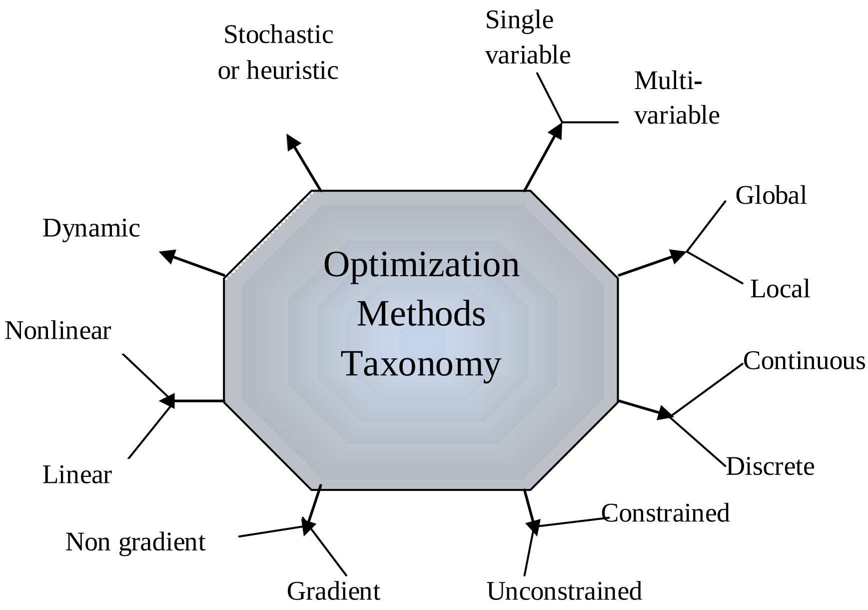


Figure 2.1: Optimization Methods Taxonomy

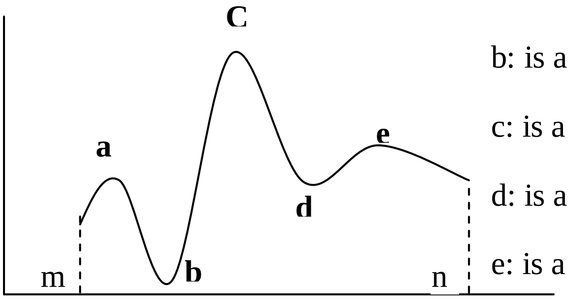
2.3.3. Classification of Optimization Methods

Many classification categories for the optimization techniques depend on a natural of the optimization problem as inputs, outputs and objective function of the problem. The main classifications of the optimization techniques are illustrated in Fig.2.1 above; some of these classifications are illustrated as follow:

A. Classical Optimization Methods

These methods are used to find the optimal solution for continuous and differentiable functions; these methods find the optimal solution analytically. These methods are classified into four categories depending firstly on the number of inputs for the function:

Single-variable methods: when the objective function has one input (one parameter), and **Multi-variable methods**: when the objective function has many inputs (many parameters).



a: is a local maximum

b: is a global minimum

c: is a global maximum

d: is a local minimum

e: is a local maximum

Figure 2.2: Global and local minima

It secondly depends on the type of the optimal values for the function:

G**lobal optimization methods**: if they always get the global optimal solution, or **local optimization methods**: if they always get the local optimal solution. Fig.2.2 above illustrates the global and the local optimal solutions for some functions [9].

And here the objective function for single variable as: $\boldsymbol{min.f(x)}$

And here the objective function for multi variable as: $\boldsymbol{min.f(x_1, x_2, \ldots, x_n)}$

Constraints: $\boldsymbol{m <= x <= n}$, $\boldsymbol{f}$ function must be continuous and differentiable.

B. Linear Programming (LP)

This is used to solve the problems in which the objective function and constraints appear as linear functions; linear programming was developed in the 1930 by economists [9].

C. Nonlinear Programming (NLP)

This is used to solve the problems in which the objective function is too complicated to solve as nonlinear functions, and is also used when classical analytical methods fail in finding the optimal solution. Some of these methods are **constrained:** when the objective function has constraints and others are **unconstrained minimization methods**: when the objective function has no constraints.

Nonlinear methods may be classified into two categories: **direct search methods** or **nongradient methods:** these methods deal with non differentiable objective function, while **decent methods** or **gradient methods** deal with differentiable objective function [9].

D. Dynamic Programming

In practical problems, decisions are taken sequentially at different levels in time or space for a component of a system or subsystem. Dynamic programming methods are mathematical approaches that are fit for the optimization of multilevel decision problems. These methods are very suitable to solve a wide range of complex problems in several areas of decision making such as those of a salesman, and shortest path problems which named greedy algorithms. The main solution concepts depend on the use of a recursive technique to solve sub- problems [9].

E. Integer Programming

Optimization methods can be classified depending on the input variable:

Continuous optimization methods: these are used when the objective function has a continuous input variables (real values) while **Integer optimization programming methods** are used when the objective function has a discrete input variables (just integer values) [9].

F. Stochastic Programming

Stochastic, heuristic, probabilistic or random programming is suitable to solve problems when the inputs' parameters of the optimization problem are random values at each run [9]. In recent decades, researchers have increasingly turned to the stochastic optimization techniques and metaheuristic optimization techniques in particular. Stochastic programming does have many positive features, but it is still controversial. As mentioned earlier, results obtained from a stochastic optimization technique are unpredictable due to randomness. The forces of stochastic programming are as follows:

1- A search process, through the use of a stochastic optimization algorithm, may miss the global optimal solution from a wide search of space.

2- Most of stochastic optimization algorithms are inspired by natural behavior, and each stochastic algorithm has at least one control parameter. The efficiency of stochastic algorithms more or less depends on these control parameters. However, the stochastic algorithms are not simple.

3- Stochastic techniques usually require more objective function evaluations to find the optimal solution, thus making them computationally more expensive.

Some important types of stochastic optimization algorithms are evolutionary methods. These methods were inspired by Darwin's theory of evolution as biological evolution mechanism.

Three mainstream evolutionary methods were used in this research: GA [22, 23], PS algorithm [13, 24, 25], and TS algorithm [26]. GA, PS algorithm and TS algorithm are selected to optimize the traffic light signals timing problem, to minimize the average of travel time *ATT* for each vehicle. Because these algorithms are global methods, and used in the previous research for solving the traffic light timing problem. The traffic light timing optimization problem is classified as a multi-variable and stochastic problem.

Each algorithm has many control parameters which affect the efficiency of the algorithm in finding the optimal solution; the main parameters are as follows:

- Generation number: Statistically a late generation is better than an earlier generation.
- Population size: This is the number of chromosomes or lists of records. The first generation in the three algorithms must be generated randomly.
- Chromosome list or individual: This is a set of genes or records which must be equal to the number of the input parameters in the optimization problem.
- Gene, particle or record: This is the input parameters' value, and usually it is real number.

The algorithms (GA, PS, and TS) can be summarized as follows:

1. Genetic Algorithm (GA)

This is a metaheuristic search that presents the process of natural evolution. It was developed by Prof. John Holland and his students in the University of Michigan in the 1960s and 1970s [22, 23]. This algorithm generates solutions for optimization problems by using techniques inspired by natural evolution, such as inheritance of the best genes to the next generation and skipping of the worst genes from the next generations, by mutation, selection, and crossover operations.

GA is used in the application of computer science, engineering, economics, physics, manufacturing, mathematics, and other fields. Each chromosome consists of number of genes which are equal to the input parameters' values to be optimized.

2. Particle Swarm Algorithm (PS)

Introduced by Kennedy and Eberhart in 1995 [13, 24, 25], as its name implies, PS was inspired by the movement and the intelligence of swarms. A swarm is a structured collection of interacting organisms such as bees, ants, or birds together. Each organism in the swarm is a particle. Each input list is a swarm which consists of input parameters (particles), and each particle has two values: position and velocity.

3. Tabu Search Algorithm (TS)

This metaheuristic method, originally developed by Glover [26], has been successfully applied to a variety of combinatorial optimization problems [27]. Each generation consists of a number of lists and each list consists of input parameters (records). The generation size must be equal to the input list size multiplied by 2.

It can be concluded that each algorithm has many different types of operations to get a new generation from the previous generation.

The differences between steps of GA, PS and TS are illustrated in Table 2.2 below.

Table 2.2: GA, PS and TS algorithm's steps

Steps	GA	PS	TS
1	Initialize first generation randomly.	Initialize first generation randomly.	Initialize first generation randomly.
2	Compute fitness value for each chromosome, and sort them.	Compute fitness value for each list and sort them.	Compute fitness value for each list and sort them.
3	Carry out the operations (selection, cross over, and mutation) on the previous generation to produce a new generation.	By update for each particle position and velocity in the previous generation, a new generation is produced.	Update the parameters or records values in the previous generation, by adding and subtracting a fixed number to each record and saving the best one in tabu list; a new generation is produced.
4	Do step 2 until the end of population.	Do step 2 until the end of population.	Do step 2 until the end of population.
5	Getting the optimal solution at the end.	Getting the optimal solution at the end.	Getting the optimal solution at the end.

2.4 Simulation for Representation of the Transportation System

A simulation system is a real system which is represented in a software program. Simulation consists of methods and applications to simulate the real systems' behavior. It is developed almost through using computer software. Simulation became more popular and powerful in recent years since computer and software are better than ever [28].

The simulation system consists of many models, and each model may consist of sub models. Each one of these can represent the construction and the work of real sub-system from the main system. Each model has inputs, processes and outputs, and the model should represent the real system to some degree. Models must be verified and validated through comparison of inputs, processes and outputs of the model with the real system. The simulation tool is a controlled system used to evaluate and improve the performance of the system by using the powerful computer system with more reduction of cost and time than the real system. A developing process for any real system has many aims such as solving the problems, improving them to be more efficient and applicable and saving money and time. But improving the process in some complex real systems directly is not practical because the developing process would be more expensive in terms of time and cost as when the input or output behavior is random or very huge and is impossible to compute in the real world. When a simulator system is completed to simulate a real system, all input parameters can be optimized by suitable optimization techniques to improve and optimize the system results and accordingly make a decision about the development of real systems [29].

Simulation models are classified as dynamic or static, discrete or continuous, and stochastic or deterministic models [28], p.9. The simulation models are widely used in traffic network to deal with shortcomings, planning and improvement of the performance of traffic network systems.

In the 1930s, many attempts were made to develop a mathematical model for traffic flow system but they did not succeeded because the traffic phenomena are nonlinear and are complex. They depend on multi interactions such as a large number of vehicles which interact in a complex way with the system laws and the psychological reactions of the human driver. Until now even a general simple mathematical theory for describing the real traffic flow system has not been found [30]. The simulation tools are the best way for accurate description of the transportation system, and all traffic simulators depend on the similarities between traffic network systems and the queuing networks. Jackson is considered the father of the queuing theory. This theory was a mathematical model for waiting time at some server or queue or line; it provides an analysis for the relations between many related servers [31]. This theory consists of three concepts: servers, queues and customers. The inputs and the outputs of queuing system are customers or vehicles with some distributions such as exponential distribution behavior.

There are three main classes of traffic flow models which can be distinguished according to the level of details of the transportation simulation systems. These modes are microscopic, macroscopic and mesoscopic models.

In the microscopic models, a high computation time is needed because every vehicle is considered as an individual particle with some characteristics such as speed, position, acceleration and driver-vehicle interaction. These are evaluated at each time from the start to the end of run simulator, because the evaluation of these parameters mathematically is very hard. However, in the macroscopic models, it is assumed that the movement of vehicles is shown as a fluid motion and this model is computationally fast because the input requirements are very simple. The last class model is called mesoscopic; it comes between the microscopic and the macroscopic models, and each lane in this model represents as one server which serves one vehicle at a time. However, the last two models are limited in representing differences in drivers' behaviors and interacting processes between vehicles on the network [2, 30].

In this research microscopic simulation, called SUMO simulator, was selected to represent the research road network of the case study [32].

Chapter three

Research Background and Details

This chapter is devoted to the main traffic light signals optimization problem and the traditional methods, used to optimize this problem, and their shortcoming. The data of the road network of the case study, definition of the main objective function of the traffic light signals timing optimization problem, the main simulation tool used to simulate road network of the case study, SUMO simulation explanation, and selection of suitable benchmark function. The chapter also dwells on other tools used to help in solving the problems like parallel processing tool. All methods were selected to be used in solving the traffic light signals timing optimization problem. These include random algorithm and metaheuristic optimization methods (GA, PS, TS).

3.1 Traffic Light Signals Timing Optimization Problem

What is the traffic light signals timing optimization problem?

To answer this question, let us assume we have 4 intersections and 13 traffic light signals on our real road network, and this network is working in a city. We have to ask another question: Are the traffic light signals on the road network working in the optimal way or not relative to waiting time and to travel time? To answer this question, some optimization processes for the traffic light signals timing parameters must be done:

1. The optimization process can use many techniques, models and sometimes simulation tools of the real world road network.

2. There is a strong relationship between the traffic light signals timing system control and the total travelling time for all vehicles in the network discussed in interactive research in the previous chapter. Therefore, by changing the period time for one or more traffic light signals time, such as green light or red light, the total travel time value for all vehicles may ascend or descend, and the best time value will be when the total travel time decreases, but this relation isn't directed or simple.

Municipality engineers use different simple approaches to find the optimal timing for the traffic light signals to minimize delay time or total travel time and to maximize throughput (number of vehicles exiting the road network). These methods are as follows:

❖ **Webster method**: This method is used to find the optimal cycle time length for isolated intersection mathematically. The green time for each lane movement is used to minimize the delay time to the global minimal [14]. The steps of this method are as follows:

1. Each lane at any intersection *i* has a saturation flow S_i, (the ideal case for this lane) When the lane has a green light for one hour, and the maximum number of vehicles is 3,700 vph, 2,000 vph or 1,600 vph, the condition and the environment at the intersection determine the saturation flow.
2. There is a lost time for each traffic light time phase L_i, which might be between 3.5s to 5s, and yellow light time ranges from 2s to 5s as we presupposed.
3. The total percentage of vehicle flow to the saturation flow for some intersections was computed by collecting all the percentages of vehicle flows, for each lane group works together to the saturation flow for this lane group on the same intersection.
4. By the Webster equation [14], chp.8, pp.(353-355), the optimal cycle length can be computed as follows:

$$\boldsymbol{C_0 = \frac{1.5 * L + 5}{1 - \sum_{i=1}^{i=\emptyset} \frac{Lv_i}{LS_i}}} \quad \dots\dots\dots\dots\dots\dots(3.1)$$

where C_0: is the optimal cycle length.

L= lost time in the cycle time which we presupposed = 4s for each traffic light time phase.

Lv_i= actual lane volume in one hour.

$\emptyset$= number of traffic light phases.

Ls_i- Saturation flow for each lane which we presupposed = 1,600 vph (in this research).

The optimal green time for each traffic light signal can be computed as follows:

$$\boldsymbol{G_t} = \left(\frac{\boldsymbol{Lv_i}}{\sum_{i=1}^{i=\emptyset}\frac{\boldsymbol{Lv_i}}{\boldsymbol{LS_i}}}\right) * (\boldsymbol{C_0} - \boldsymbol{L}) + \boldsymbol{l_i} - \boldsymbol{Y_t} \quad \text{.................(3.2)}$$

Where G_t: the actual green time phase for the traffic light signal.

Lv_i= actual lane volume in one hour.

Ls_i= saturation flow for each lane 1,600 vph, (presupposed).

C_0= is the optimal cycle length.

L= lost time in the cycle time which we presupposed = 4s for each traffic light time phase.

L_i= lost time for the green phase=4s.

Y_t= yellow time phase for the traffic light time which we presupposed = 3s.

❖ **HCM method**: This method was used to determine the optimal cycle length and green time, for isolated intersection mathematically, to minimize the delay time to the global minima [14]. This method is close to Webster method and is used to determine the optimal cycle time length and green time for each traffic light signal to minimize the delay time for the global minima. The main equations are as follows [14], chap.8, p.(356):

$$(\boldsymbol{v}/\boldsymbol{c})_{\boldsymbol{i}} = \mathbf{x}_{\boldsymbol{c}} = \sum_{i=1}^{i=\emptyset}\left(\frac{\boldsymbol{Lv_i}}{\boldsymbol{LS_i}}\right) * \left(\frac{\boldsymbol{C}}{\boldsymbol{C-L}}\right) \quad \text{...........(3.3)}$$

Where x_c= is the critical v/c ratio for the intersection.

$\sum_{i=1}^{i=\emptyset}\left(\frac{Lv_i}{LS_i}\right)$= summation ratios of actual flow to saturation flow for all lanes.

C= is the optimal cycle time length in s.

L= is the total lost time in the cycle time which we presupposed = 4s for each phase.

The green time equation as in the eq.(3.2).

❖ **SYNCHRO simulation program**: Developed in the US to optimize the traffic light signals timing for computing many objective functions [15], this program is a macroscopic simulation, is very simple, and is widely used in the world [2]. Nablus municipality is using it to optimize the traffic light signals timing parameters on its road network.

By using Webster and HCM methods, the optimal cycle time length and green time for each traffic light signal in the road network of the case study, illustrated in next section, will be computed to minimize the *ATT* for each vehicle to the global minima. Table 6.1, chapter 6, shows results of each optimal time list of Webster, HCM methods and SYNCHRO simulation.

However, the main shortcoming of these methods is that they could be just applied on isolated intersections, and when the flow rate is small or a large, these methods are ineffective as the phase time is not logical, as when the green or red phase time =2s or 250s. In this context, we ask the main research questions:

Do these methods (Webster, HCM and SYNCHRO) determine the real optimal timing for each traffic light signal in our case study road network? And could the delay time at the traffic light signals be global minima?

To check the previous conclusion and to answer these questions, the researcher optimized the traffic light signals timing parameters by using different metaheuristic optimization techniques (GA, PS and TS) to find the optimal timing for each traffic light signal. The research case study road network, as a global way, (ex. all traffic light signals at the same time), was used to minimize the average travel time (*ATT*) for each vehicle to the global minima or close to the global minima. Then the results of these algorithms are compared with the results of Webster, HCM and SYNCHRO. Here another question could be raised:

Can we find the most suitable metaheuristic optimization algorithm that arrive at the optimal solution or more efficient solution,(ex. closer to the optimal solution) for the traffic light signals timing problem, out of the space solution, to minimize the average travel time for each vehicle?

In this research, each traffic light signal time (green or red), as we presupposed, can take a time value from 10s- to- 100s. In the road network of the case study of this research in the next section, there were 13 traffic light signals which might have different light colors (green and red) at the same time. Therefore, each traffic light color period can take a value between 10s - 100s. The 13 traffic light signals have a huge space as shown in Table 3.1 below.

There are also many symbols:

TL= is a one traffic light signal time.

TL_ir = means the time of red light for traffic light signal *i*.

TL_ig = means the time of green light for traffic light signal *i*.

T= is a time list of 13 records of traffic light signals time.

The main time list is T= [$TL1$, TL_2, TL_3, TL_4, TL_5, TL_6, TL_7, TL_8, TL_9, TL_{10}, TL_{11}, TL_{12}, TL_{13}].

The time list contains 13 records which represent the traffic light signals time phases as: T= [TL_1g, TL_2g, TL_3g, (TL_4g, TL_4r), (TL_4r, TL_4g), (TL_5g, TL_5r), (TL_5r, TL_5g), (TL_6g, TL_6g, TL_6r), ($TL6r$, TL_6r, TL_6g), (TL_6r, TL_6g, TL_6g), (TL_7g, TL_7g, TL_7r), (TL_7r, TL_7r, TL_7g), (TL_7r, TL_7g, TL_7g)]

Table 3.1: Traffic light signals timing probabilities.

Traffic light* time	Time				
	t_1	t_2	t_3	…	t_{90}
*TL*1	10s	11s	12s	…	100s
*TL*2	10s	11s	12s	…	100s
….	…	…	…	…	…
*TL*13	10s	11s	12s	…	100s

Suppose that each green or red light time period were $10s \leq TLig$, $TLir \leq 100s$, and each traffic light signal time had 90 probabilities, then the solution space would be as follows:

The number of probabilities for the solutions grows up. Exponential 90^{13} means:

$$(\text{Number of each time value probabilities})^{(\text{Number of traffic light input values})}.$$

3.2 Case study

The road network, used in this research, is located in an important city center: Nablus city. The streets and the junctions had heavy traffic at peak hours. At peak hours there was always traffic congestion, especially in the municipality building area. The schemes and measurements of road network parameters used in this research were taken from the Engineering College of An-Najah National University. The flow of vehicles at each traffic light signal data is shown in Fig. 3.1 below. The figure represents the historical real data of vehicle movement collected on September 21, 2010 for one hour between 7:00 am and 8:00 am at each traffic light signal.

The total number of vehicles which entered the road network = 1,640 vehicles. The researcher presupposed that all vehicles would have the same type in this research.

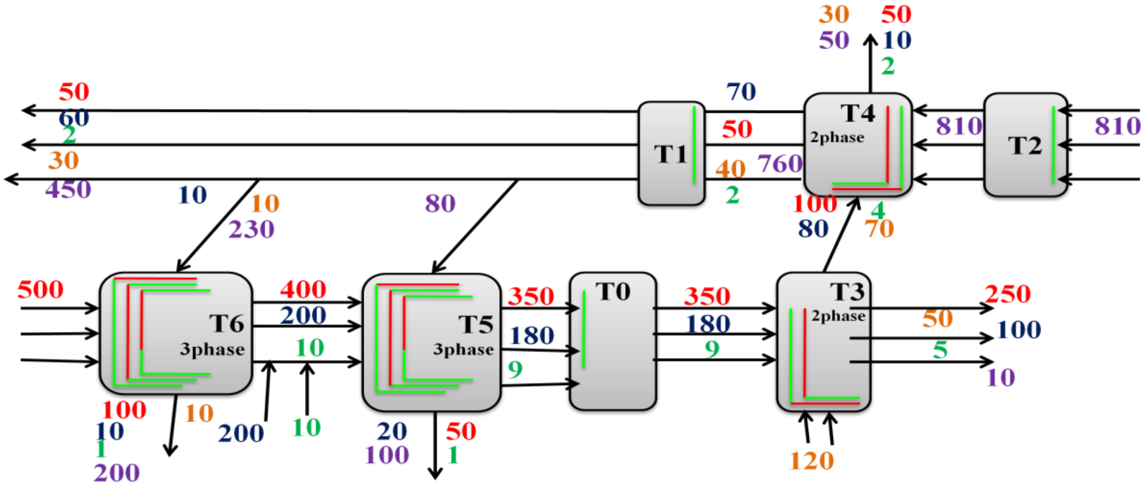


Figure 3.1: Road network case study with vehicle volume per hour at each traffic light signals collected September 21, 2010.

The phases of all traffic light signals in the case study are represented as the time list T= [TL_1, TL_2, TL_3, TL_4, TL_5, TL_6, TL_7, TL_8, TL_9, TL_{10}, TL_{11}, TL_{12}, TL_{13}].

T= [TL_1g, TL_2g, TL_3g, (TL_4g, TL_4r), (TL_4r, TL_4g), (TL_5g, TL_5r), (TL_5r, TL_5g), (TL_6g, TL_6g, TL_6r), ($TL6r$, TL_6r, TL_6g), (TL_6r, TL_6g, TL_6g), (TL_7g, TL_7g, TL_7r), (TL_7r, TL_7r, TL_7g), (TL_7r, TL_7g, TL_7g)]

3.3 Traffic Light Signals Timing Model and Objective Function

The objective function equations below show the complexity level of the traffic light signals timing optimization problem which this research is trying to solve as a total way. The objective function (or fitness function) for optimizing the traffic light signals timing parameter is minimizing the average travel time for each vehicle, which leads to minimization of delay time at the red traffic light signals and minimization of the total co_2 gas pollution which comes from the stopping vehicles.

In this research, the researcher presupposed the period time of each traffic light signal phase time *TLir* or *TLig* ϵ {10s, 11s, 12s, …, 100s}, and each traffic light signal time phase has 90 probabilities times in seconds. Time of yellow light phase is constant *TLiy* =3s, for all traffic light signals.

The main objective function is: *Min. f(T)= Min.*(average travel time (for each vehicle)).

These symbols stand for the following meanings:

1- *TTT*= Total Travel Time for all vehicles (*s*).

2- *ATT*=Average Travel Time (*TTT*/vehicle) for each vehicle in *s*.

3- *TT*= Travel Time for each vehicle in *s*.

4- *V*= average velocity of the vehicles (*m/s*).

5- *Li*= Length from start point to distention point for each vehicle in meter.

6- *n*= number of vehicles in the road network .

7- $V_1(t)$= speed of the leading vehicle in time (*t*) which changeable with time.

8- *g(t)*= gap to the leading vehicle in time (*t*).

9- Ʈ= is the driver’s reaction time (usually 1s).

10- $b(t)$= the deceleration function.

11- $TTi= (L/V)$, for each one vehicle from start point to end point.

12- $vi(t)$= vehicle speed at some time (t)

13- t = a time in seconds.

14- TL_i = a time in seconds for the traffic light $_i$ phase.

The main equations for the objective function (*ATT* for each vehicle) are as follows:

$$f(T)=ATT$$

Cost function: $$\boldsymbol{Min.\ f(T)=Min.(ATT)}\ ....(3.4)$$

Constraints

Where T= $[TL_1, TL_2, TL_3, TL_4, TL_5, TL_6, TL_7, TL_8, TL_9, TL_{10}, TL_{11}, TL_{12}, TL_{13}]$.

$$TL_i \in \{10s, 11s, 12s, 13s, \ldots, 100s\}.$$

****

And the *ATT* for each vehicle calculated as: $\boldsymbol{ATT = TTT/n}$............(3.5)

From movement equations: $$\boldsymbol{TTT} = \sum_{i=1}^{i=n}(TT_i) = \sum_{i=1}^{i=n}\left(\frac{L_i}{V_i}\right)........(3.6)$$

The travel time for each vehicle is calculated by SUMO simulator which depends on a car flowing model [33]. And the vehicle's speed logic in this model is adapted to the speed of the leading vehicle, and the drivers should lead their vehicles in a safe velocity. Therefore, each vehicle may be driven with a changing speed at each changing time by depending on the safety distance between the vehicles, the speed of vehicles on the road, the traffic light signals control, and the scenarios of movement of vehicles (stopping, fixed velocity, acceleration, and deceleration). Below is a car-flowing model which explains the vehicles motion mechanism like in SUMO simulator steps working [33], as in eq.(3.7):

$$V_{safe(t)} = v_i(t) + \frac{g(t) - v_i(t)\tau}{\frac{V}{b(V)} + \tau}.........(3.7)$$

By replacing (vi) in eq.(3.6) with $V\ safe(t)$ in eq.(3.7), eq.(3.8) will exist to explain how the total travel time will be computed.

$$TTT = \sum_{i=1}^{i=n} \sum_{t=0}^{t=end} \left(\frac{L_i}{v_i(t) + \frac{g(t) - v_i(t)\tau}{\frac{V}{b(V)} + \tau} v_i(t) + \frac{g(t) - v_i(t)\tau}{\frac{V}{b(V)} + \tau}} \right) \ldots\ldots.(3.8)$$

So when traffic light is red, the velocity of each vehicles on this traffic light signal is zero, so each vehicle crosses a number of traffic light signals through its trip, and the velocity of the vehicle =0, upon arrival at the red traffic light, but:

* The driver cannot know when the vehicle would arrive at each traffic light signal, and whether the traffic light signal would green or red.

* Each vehicle has its own number of traffic light signals crossed.

From these equations, and depending on the logic of traffic light system, when the velocity of vehicles =0, it means vehicles would always stop on the red traffic light, and the vehicles would accelerate their speed when the traffic light is green. So to minimize the *ATT*, we can optimize the traffic light signals timing.

Traffic light signal systems and road network systems and the optimization process for traffic light signals timing parameter, as a global way, are very complex because the mathematical relation between the inputs (13 traffic light signals timing) and the outputs (average travel time) is not definite, and the topology of this relation cannot be known, because it has huge probabilities and random behavior. Therefore, the best way to get the global optimal solution is by testing all inputs probabilities (90^{13}). Then the best input that gives a minimum result of the average travel time for each vehicle would be selected. But the question is how implemented this way.

First, in the real world, we have thousands of vehicles, and each one has its own parameters which change relatively in terms of time, speed, position, direction, drivers' reactions and the traffic light signals phases in the city. Therefore, we cannot implement and optimize this problem in the real world. It is very difficult, complex, expensive, and time consuming since we depend on a hundred millions of probabilities which take a very long time: years.

Second, some researchers think that using computer systems to implement the road network system is very efficient because the computational efficiency is good to solve the previous problems: cost, complexity and capacity.

Also the simulation systems are the most suitable way to implement a real complex system problem.

In this research, a random algorithm and metaheuristic optimization techniques (GA, PS and TS) were used to optimize the traffic light signals timing problem, and to compare the one that has the most suitable results.

3.4 SUMO Simulator

Traffic simulation models can be helpful in estimating/predicting the conditions such as delays, travel times, queues and flows. These models can predict future conditions and can be used to optimize network operations for current and future real world conditions. SUMO simulation is a famous tool and is widely used for studying random and complex real-world systems.

SUMO simulator was developed in 2000 [32]. The major reason for the development of an open source microscopic road traffic simulation was to support the traffic researchers with a tool and ability to implement and evaluate algorithms. This tool has no need to consider all the necessary things to obtain a complete traffic simulation such as implementing and/or setting up methods to deal with road networks, demand, and traffic controls.

SUMO simulator has input files and output files, thus generating just one output file is named tripinfo.xml. This file at the end of running for the simulator will contain the total data about each vehicle trip on the SUMO simulator. This data includes vehicle's id, type, starting and ending time, arrival speed, lane, and lane length. The three input files are named new.sumo.cfg, new.net.xml, and rout.rou.xml. Net.xml and rout.xml have data about network building and control such as lanes, nodes, positions, maximum speed, logic, traffic lights, junctions, vehicles routes and priority while sumo.cfg file has a configuration file to join input files and output file with SUMO simulator.

The road network of the case study, which was simulated, has 13 traffic light signals as main factors. This research studied the effect of these parameters' values on the average travel time for each vehicle to get the optimal values for these parameters which produced the minimum of the average travel time for each vehicle. The steps of SUMO simulator, were used in this research with each optimization method, are as follows**:**

1- When SUMO simulator runs to control the traffic light signals on the road network, there are 13 time values. A time list had to be set as input list, and each time value binds to one traffic light signal case. To that end, a python code function was written to input the time list to the new.net.xml file before running the simulator:

Define integer time list[13]=[20,30,40,55,45,39,67,81,49,58,15,72,69]
//set each value randomly which represents one traffic light time.
{Open new.net.xml file
Write the list on it in traffic light signals time location
Close new.net file
Call sumo simulator with new.sumo.cfg file}

2- The results were saved in some files. Then the time list input was changed into a new list: [78,67,56,75,46,33,24,42,35,93,84,61]. The simulator was run another time; then the new results were saved, and these steps, many times, were iterated. At the end, the results differed according to the input time list. The average travel time for each vehicle might be up or down according to the traffic light signals time list. The number of vehicles which stopped on the red traffic light signals and the average travel time for each vehicle might be up or down. The best traffic light signals time list when the result is the minimum value for the average travel time for each vehicle.

3- To get the minimum average travel time for each vehicle, there is a very big number of different traffic light signals lists. Time inputs must be tested and a simulator be run with each one. The best list time has a global minimum of the average travel time result for each vehicle. This process is named an optimization process. But how much time of running simulator with different list time do we spend to get the optimal list?

4- In this research, each traffic light signal time case can take a value between 10s and 100s, so each record in the time list has 90 probabilities, and the list time consists of 13 records. The number of different time lists which we must try, to get the optimal traffic light signal list time, is 90^{13}, and this is a very huge number: about thousands of millions or trillions together. Each running of SUMO simulator for one hour takes about 30s. Therefore, to get the optimal time list, many years of running SUMO simulator would be needed, and to get the global minimum of fitness function, all probabilities input time lists must be tested. This solution is neither efficient nor applicable. *Solving this problem is a big challenge*. For this reason, many metaheuristic optimization techniques had been clarified in this chapter and were used to get accurate and nearest solution to the optimal one.

5- Each algorithm started with random selection for the first generation of the time lists. Then the simulator was run for each time list, and the average travel time for each vehicle was computed and saved. After that, the lists of results were sorted, and by the algorithm's operations, the new generation would be built depending on the previous generation.

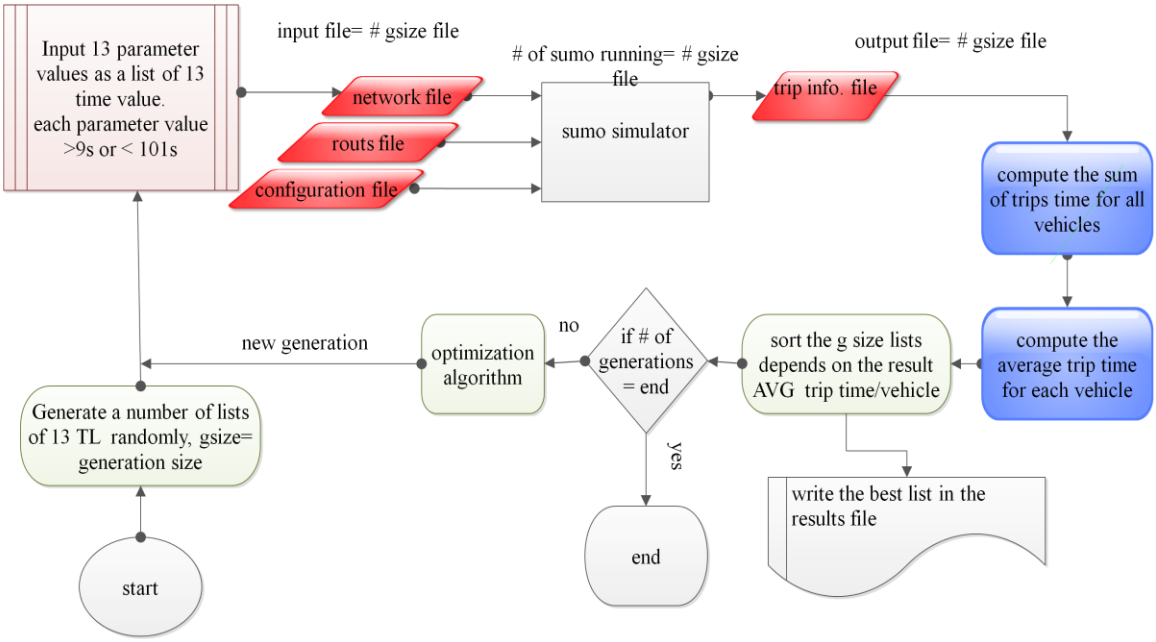


Figure 3.2: Schema of SUMO simulator and optimization algorithm steps and parts.

This step was replicated many times until the end of the generation numbers were gained. The schema in Fig.3.2 above illustrates the used steps and parts.

3.5 Benchmark Function Selection

The function of the traffic light signals timing is of an unknown topology. After trying some metaheuristic optimization algorithms with SUMO simulator in the first experiment, the results couldn't decide if the optimization algorithm was the best or not, because the optimal solution is unknown and the results could not decide whether the best solution was the optimal solution or near the optimal solution or far from it. Therefore, the researcher decided to get some suitable benchmark functions to test optimization algorithms used.

Some benchmark functions were found in [21]. This function has many local minimum values and the global minimum value is known; the number of input parameters is equal to the number of input parameters in traffic light signals timing problem. The steps of benchmark function used the same steps in the traffic light signals timing problem just by replacing the SUMO simulator with a benchmark function. And the schema of used steps of the benchmark function is illustrated in Fig. 3.3 above. The benchmark function is called Rastrigin’s function [20]:

$$y = 10n + \sum_{i=0}^{i=n}[xi^2 - 10\cos(2\pi xi)] \ldots\ldots\ldots\ldots\ldots(3.9)$$

$-5.1 < xi < 5.1$, and the minimum when y=0 at $x = 0$

* n: is the number of records in each list =13 records, and it is equal to the records of time list in traffic light timing problem.

The range of probabilities of each input parameters is also equal or close to the range of probabilities of the input parameter in the traffic light signals timing problem presupposed in this research. As xi: is the input record value in each list, and $-5 < xi < 5.1$, the different input list of values are 102 probabilities, and these probabilities are near from the number of input in the traffic light signals timing experiment which takes 90 probabilities’ values.

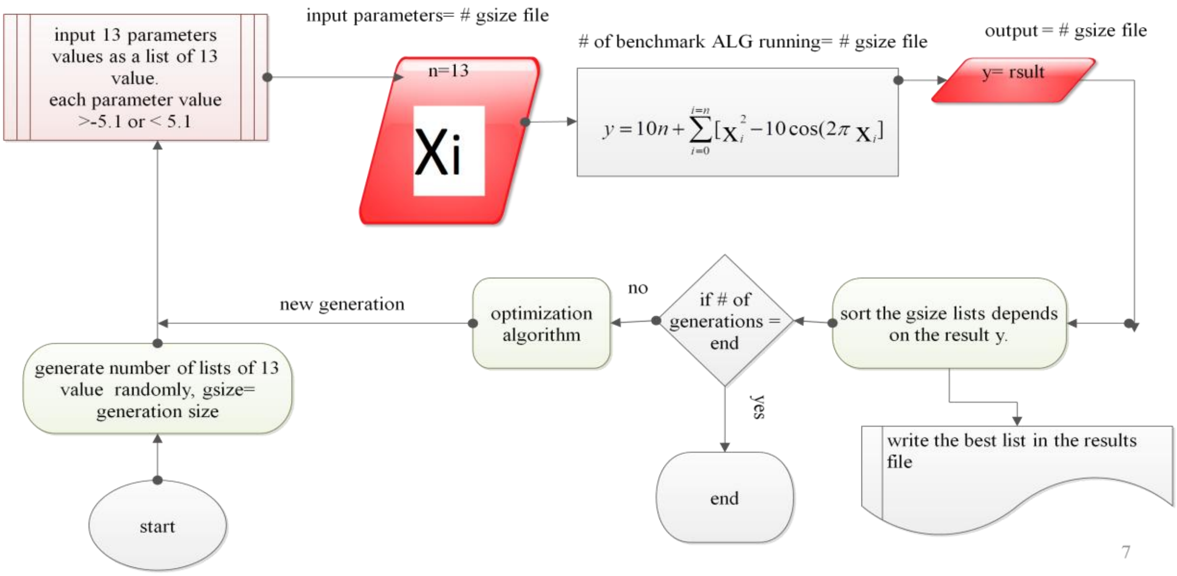


Figure 3.3: Schema of benchmark function and optimization algorithm steps.

❖ Notes: There is no relation between the benchmark function optimization problem and the traffic light signals timing problem. Therefore, if an optimization algorithm is the best in solving the benchmark function, it may not be the best in solving traffic light signals timing problem and the opposite is true.

Benefits of using the benchmark function in this research:

1- To determine the efficiency of the optimization algorithms and find the optimal solution when the results are minimum.

2- To help us in finding the best conditions for the metaheuristic optimization algorithms were used to solve the traffic light signals timing problem. Through experiments the significance of this function was illustrated in the speed of finding the best conditions for the algorithms in a short time.

3.6 Parallel Processing

SUMO simulator running time for each chromosome (ex. time list) takes about 30 seconds on one computer. For example, in Experiment 1, in the next chapter, each generation takes about 15 minutes to end running. Therefore, to finish 50 generations on one PC, it may take about 12 hours running time, and this time is very long. In this research, a parallel processing was used to reduce the running time for each generation by using a parallel processing server with python language (named: ppserver) on one computer with 4 or 8 processors [34]. That means 4 or 8 running simulators at the same time. In this case, the generation time was reduced to ¼ or more, as running 50 generations in (30 ch.*50 gen.* 30s)/4≈ 3 hours. Implementing this way on more than one computer could be done in this research because a cluster had to be found to do that but there was no cluster available.

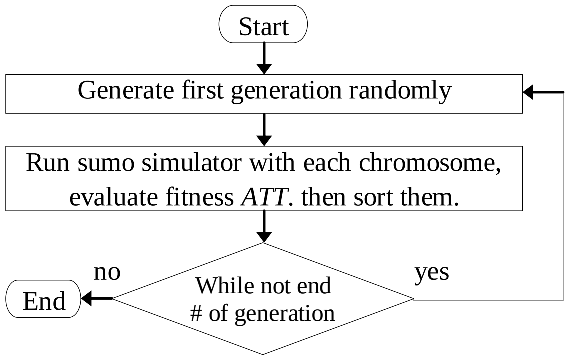


Figure 3.4: Random algorithm flowchart

3.7 Random Algorithm

In this algorithm, each chromosome in all generations was selected in a random way without a optimization process. The flow chart in Fig.3.4 above shows the steps and the operations of this algorithm. This algorithm was used in this research as a base algorithm to compare its results and the results of the metaheuristic algorithms were used in this research to see if the metaheuristic optimization algorithm was good or not.

3.8 Metaheuristic Optimization Techniques

Metaheuristic optimization techniques (GA, PS and TS) were used in this research to optimize 13 traffic light signals timing (in a global way) found in the road network of the case study to minimize the average travel time for each vehicle. The conventional optimization methods are not applicable to optimize the traffic light signals timing problem, "… These methods require a closed form function to find the optimal values directly"[8]. And the traffic light signals timing problem do not need to be represented in a close function, because of the complexity and random system. Therefore, the engineers optimized the traffic light timing problem for each isolated intersection using Webster, HCM methods and SYNCHRO simulation.

In this research, the researcher presupposed that the metaheuristic optimization methods were very suitable to optimize the traffic light signals timing parameters (global way) with the microscopic models. GA, PS and TS algorithms, in addition to many types for each algorithm, were used in this research with a tuning process. Benchmark function was used for each algorithm's parameters' values (population size, generation size, swarm size, *w*, *cp*, *cg*, *tau* and *k* add/drop, selection, crossover and mutation rate) to determine the best parameters' values for the most efficient algorithm to get optimal or near optimal solution in solving traffic light signals timing optimization problem. These algorithms' types are as follows:

3.8.1. Genetic Algorithm (GA)

GA, which consists of natural selection and genetics, is a global evolution optimization search tool. It depends on the concept of biological evaluation. This algorithm in general starts from a random population [2, 12, 22]. After candidate solutions are evaluated, selection process, cross over and mutation are applied on the previous generation to get a new generation. The best solutions have more probabilities in producing new generations.

The three types of GA were used in this research are:

3.8.1.1. GA Type 1: In this type of GA, the best individual (ex. Chromosome is a time list) may not be selected to continue in the new generation, but it has much diversity to get good solutions when the number of individuals and generations is large [12, 35].

The algorithm steps and operations are illustrated in a pseudo code:

a. *Randomly generate the first generation of individual potential solutions.*
b. *Evaluate the objective function ATT, for each generation record.*
c. *While not (number of generation reached):*
    1. *Select two chromosomes individuals randomly (two times).*
    2. *Average crossover of best two selected individuals to get a new one.*
    3. *Mutate with the probability, randomly mutate the output of previous step.*
    4. *Repeat steps (1, 2 and 3) until new generation completed, and then return to b step.*

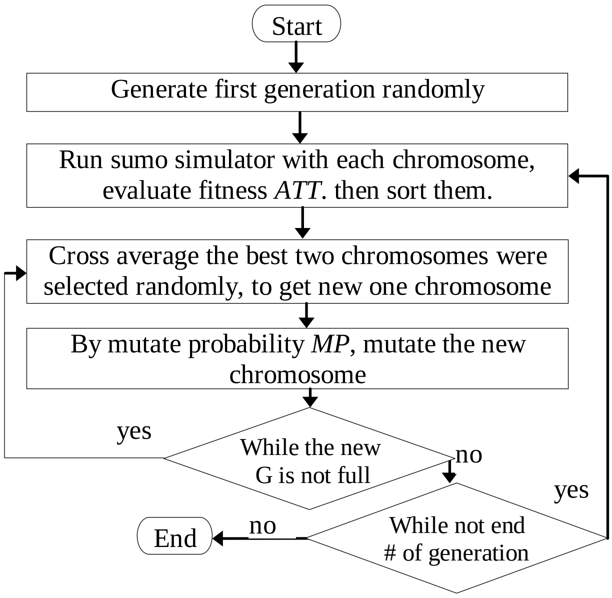


Figure 3.5: GA Type 1 algorithm flowchart

The algorithm steps and operations are shown in Fig.3.5 above.

Operations of this algorithm are as follows:

1. Selection: Select randomly two individuals (two times). Then the best two will survive and cross average.
2. Cross operations: cross average the selected individuals $(\frac{p_{i1}+p_{j1}}{2}, \frac{p_{i2}+p_{j2}}{2}, \ldots, \frac{p_{in}+p_{jn}}{2})$.
3. Mutation: with probability *MP* mutate the new individual, by Change one parameters by randomly selecting a value out of its feasible range (like 10s to 100s).

3.8.1.2. GA Type 2: This type of GA was inspired by the mated queen of bee; the best two individuals from each generation would be selected as parents [7]. Then by crossover and mutation between the parents, two new offspring would be obtained, and other individuals would be generated randomly. The algorithm steps are illustrated in the pseudo code:

a-Randomly generate the first generation of individuals' potential solutions.
b-Evaluate the objective function ATT, for each generation record.
c-While not (number of generation reached):

1. *Select the best two chromosomes from previous generations as parents.*
2. *Crossover between selected chromosomes to get a new two offsprings.*
3. *Mutate all parents and with the mutation probability; mutate offspring.*
4. *Generate randomly the other chromosomes until new generation completed, and then return to b step.*

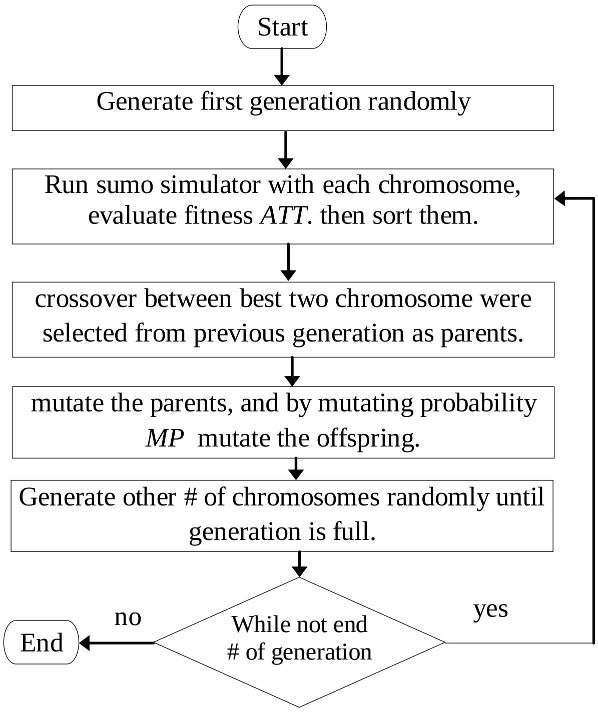


Figure 3.6: GA Type 2 algorithm flowchart.

The algorithm steps are shown in Fig.3.6 above. The operations of this algorithm type are as follows:

1. Selection: The best two chromosomes from previous generations would be selected.

2. Cross-over: First two chromosomes are taken and selected as the best. Crossover between them is made to get the two new chromosomes as offsprings:

   Chromosome1= {t1, t2, t3,… ,tn} and select a value= *c* between 1-to-n randomly,

   Chromosome2= {r1, r2, r3,… ,rn} then crossover from c point**:**

   Chromosome3 = {t1,t2,…, tc,rc+1, rc+2,…, rn}.

   Chromosome4 = {r1, r2, …, rc, tc+1, tc+2, …, tn}.

3. Mutation: Mutate the first two chromosomes (parents), and then with probability *MP* mutate the offspring.

3.8.1.3. GA Type 3: In this type of GA, the best fitness of half number of individuals is selected as parents; the worst half is discarded [35]. Then by crossover, operations between each two parents are selected to get a new two individual offspring.

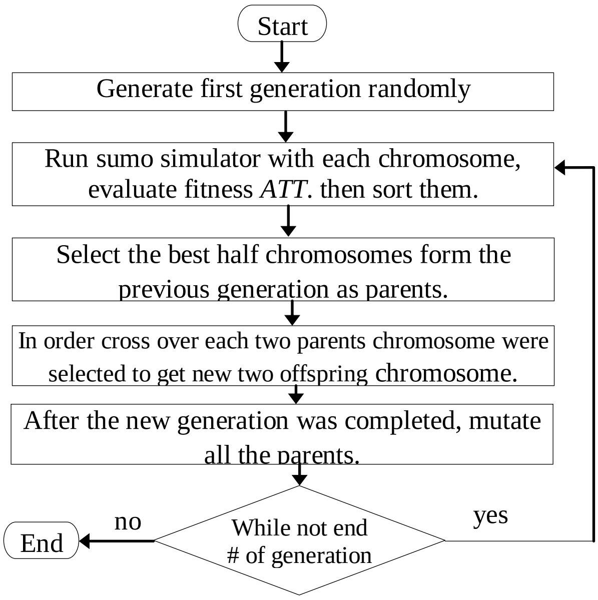


Figure 3.7: GA Type 3 algorithm flowchart

The steps and the operations are shown in Fig.3.7 above in a pseudo code:

a- Randomly generate the first generation of individuals' potential solutions.
b- Evaluate the objective function ATT, for each generation record.
c- While not (number of generation reached):
1.Select the best half chromosomes from previous generations as parents
2.Crossover between each two parents in order to get two new offsprings.
3.Mutate all the parents.
4. New generation has been completed, return to b step

Operations of this algorithm type

1. Selection: select the best half chromosomes from the previous generation as parents.
2. Cross-over: Crossover between each two chromosomes is selected in order to get the other half of the new chromosomes as offsprings and complete the new generation:

   Chromosome 1= {t1, t2, t3,… ,tn} and select a value = *c* between 1-to-n randomly,

   Chromosome 2={r1, r2, r3,… ,rn} then crossover from *c* point**:**

   Chromosome 3={t1,t2,…, tc,rc+1, rc+2,…, rn}.

   Chromosome 4={r1, r2,…, rc, tc+1, tc+2,…, tn}.
3. Mutation: Mutate the parents' chromosomes by random selection of one parameter (record), and select randomly value (like between 10s to 100s). After that, set it in the record selected.

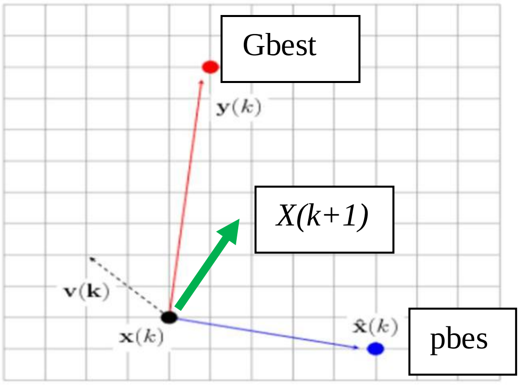


Figure 3.8: Particle swarm behavior

3.8.2. Particle Swarm Algorithm (PS)

PS algorithm is a global evolution optimization method based on the social behavior: movement of fish, birds and bees. Each particle at the beginning starts from a random position and velocity parameters' values [12]. Then, the particle direction and velocity change to move between their movement (*w*) and the best neighborhood particle movement (*cp*) and the global best particle movement in the swarm (c*g*). Fig.3.8 above shows a pointed line at *v(k)* which represents the particle movement; the best neighborhood particle movement is at *Pbest* and the global best particle movement at *Gbest*. Then *X(k+1)* represents the update movement of the particle *v(k)*.

These parameters will be tuned by benchmark function to optimize the traffic light signals timing problem, by PS algorithm, to get the optimal or near optimal timing list, which has a *min.(ATT)* for each vehicle.

The PS algorithm steps are illustrated in the pseudo code:

a. Randomly generate the first generation of individuals' potential solutions (φ).
b. Evaluate the objective function ATT, for each generation record
c. While (particle x in φ):
  1 . For each dimension I :
    *Xvi = w * Xvi + cp * random *(pbest I - Xi) + cg * random* (gbesti - Xi)// speed calculation*
    Xi = Xi + Xvi // to update the particle position
  2 . While stop step is not reached, go to the next generation.

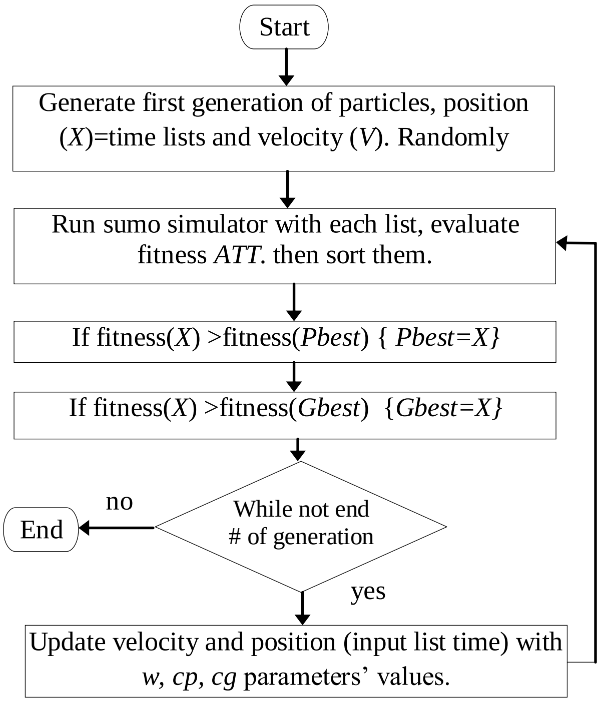


Figure 3.9: PS algorithm flowchart.

Where:
Xvi: dimension *I* velocity component

X: dimension *I* of particle position

Random: uniformly distributed random variable [0, 1].

gbest: global best value found in dimension *i*

pbest: neighborhood best value found in dimension *i*.

The PS algorithm steps are shown in Fig.3.9 above. In this algorithm, *w*, *cp* and *cg* parameters have an effect on the convergence of the particle movement to arrive at the global optimal particle [12]. The best values for these parameters are not known, and the natural selection of the optimization problem will determine the best parameters' values, but almost all the parameters' values were in this range as we see in *w*, *cp* and *cg* ϵ [0, 5].

3.8.3. Tabu Search Algorithm (TS)

TS is a global optimization technique; it has a strategy to explore new areas of solution space. At the same time, it is used to solve complex optimization problems such as timetabling, traveling salesman and so on. The main steps of this algorithm start from random generation, fit each list and sort them, generate a tabu list with maximum number of times= *tau*, to avoid reversal movement for *tau* times to the previous solutions' direction arrived at, then generate a new generation by adding/dropping *k* number to each record in the best solution. Then fit each list in the new generation aspiration. If the best list result in the new generation were more suitable than the previous best solution, then it would return to generate a new generation again from the best solution. And the new best solution is accepted if it hasn't previously been categorized (ex. Memorized) as a tabu, or the next solution is accepted as a best solution for a new generation [12, 26, 27].

k and *tau* parameters' values will be tuned by benchmark function to optimize the traffic light signals timing problem by TS algorithm to get the optimal or near optimal timing list, which has a *min.(ATT)* for each vehicle.

In this research, the standard TS algorithm was used in the first type (TS 1) to optimize the benchmark function and the traffic light signals timing problem to obtain the global minimum of the average travel time (*ATT*). Then the researcher tried to develop this algorithm by updating the start best solution in each stage and by trying different add/drop *k* values, which means the distance of neighborhood movements, as fixed or changed number, to improve the results of this algorithm.

The algorithm types (TS 2, TS 3, TS 4 and TS 5) were tried to get more suitable results than the results of the basic TS algorithm (TS Type 1) in solving traffic light signals timing problem. These types were not used or tried before this research according to the researcher's best knowledge. At the end, the developed types (TS 3, TS 4 and TS 5) produced better results than the basic TS algorithm in both benchmark and traffic light signals timing problem. These types are as follows:

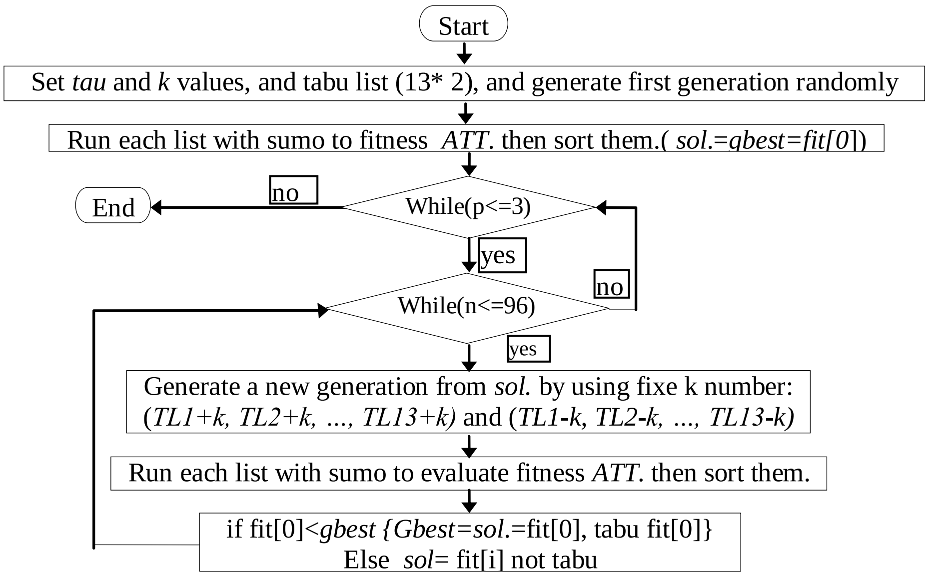


Figure 3.10: TS Type 1 algorithm flowchart

3.8.3.1. TS Type 1 Algorithm (TS 1): This is the basic TS algorithm where the tabu list consists of 26 records= (input timing parameters (13 traffic light signals time) * 2). It also consists of the number of times (*tau* value) for each previous best solution to allow reversing to the previous solution direction after *tau* value times [12].

All tasks are divided into two loops (stages). The first loop has 3 times, while the second loop has 96 times. Each step of these loops generates 26 list time as a new generation by adding and dropping a fixed *k* value which ranges from 1 to 25; *k* is the distance from the neighbors, and by comparing the previous best solution and the new best solution, the inspiration is done or not [26]. In this algorithm, the reversal step to the previous best solution direction cannot be allowed until it has arrived at *tau* times. The steps of this algorithm are illustrated in Fig.3.10 above in the pseudo code:

- *a- Determine tau and k value s, generate the tabu list with (time list records *2)record*
- *b- Generate randomly the first generation and run to determine the gbest one as a solution sol.*
- *c- First loop while not reaching repeat number=3:*
  - *Second loop while not reaching last g=96:*
  - *1- Generate new generation which contains (time list records *2) lists, by increasing and decreasing the* ***fixed number*** *(k) value to the records in the sol. list.*
  - *2- Run the new generating with SUMO to determine the best time list.*
  - *3- If (best < gbest): best is tabu and gbest = best, sol= best.*
  - *4- Else: if (best is not tabu): sol= best, else: sol =next list in generation.*

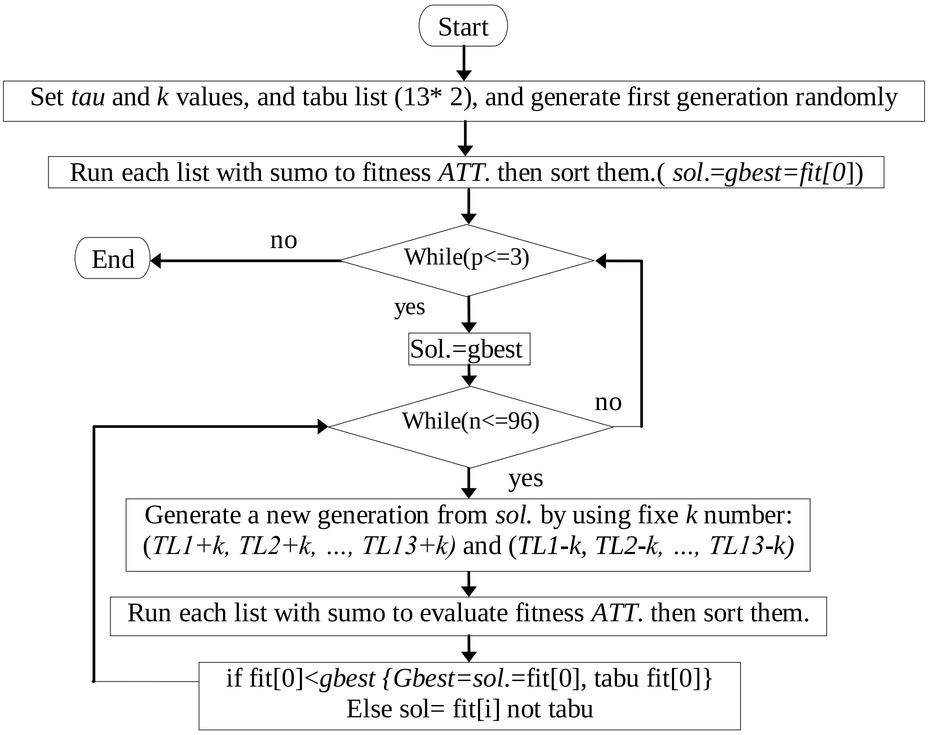


Figure 3.11: TS Type 2 algorithm flowchart

3.8.3.2.TS Type 2 Algorithm (TS 2): This type of TS algorithm is similar to the TS1 in all steps, but the reversing process for the best previous solution must be enforced at the beginning of the first loop to avoid worst solutions regions through the searching process. The steps of this algorithm are illustrated in Fig.3.11 above, and in a pseudo code:

*a- Determine tau and k value s, generate the tabu list with (time list records *2)record*

b- Generate randomly the first generation and run to determine the gbest one as a solution sol.

c- First loop while not reaching repeat number=3:

Sol= gbest

Second loop while not reaching last g=96:

1. *Generate a new generation which contains (time list records *2) lists, by increasing and decreasing the **fixed number** (k) value to the records in the sol. list.*
2. *Run the new generation with SUMO to determine the best time list.*
3. *If (best < gbest): best is tabu and gbest = best, sol= best.*
4. *Else: if (best is not tabu): sol= best, else: sol =next list in generation.*

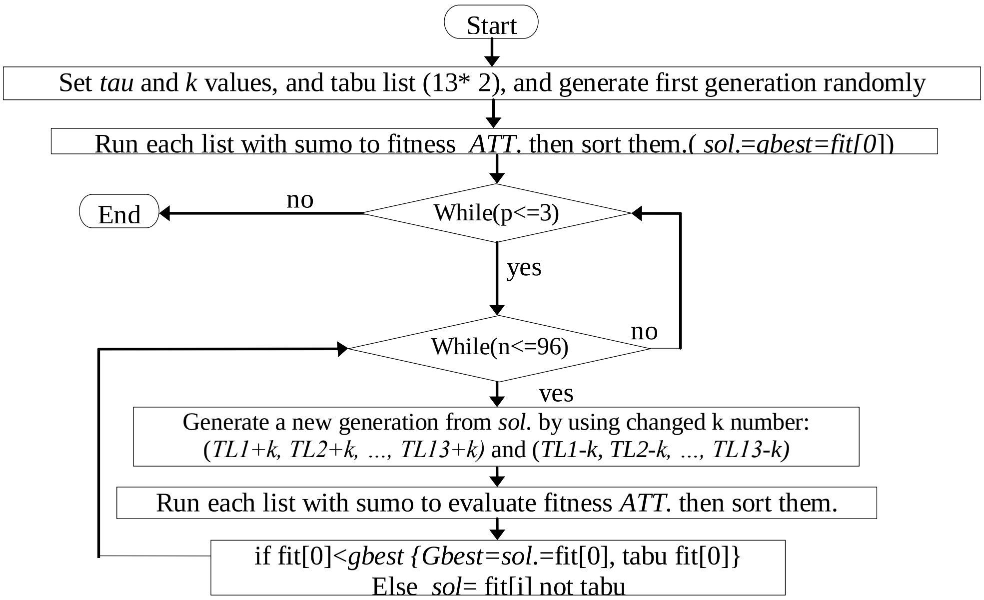


Figure 3.12: TS Type 3 algorithm flowchart

3.8.3.3. TS Type 3 Algorithm (TS 3): This type of TS algorithm is similar to TS1 in all steps, but the distance movement to the neighborhoods' *k* number changes. In the TS 1 and TS 2, the search process starts from near neighborhoods, but here the researcher presupposed that each number in the second loop *k* would change for many generations. *k* starts from a big number then it changes to a small number like [45, 40, 35, 25, 22,…, 1].

The improvements of this algorithm allow searching processes in the different regions of the solution space, as it starts from the farthest neighborhoods and then end at the nearest neighborhoods. The steps of this algorithm are illustrated in Fig.3.12 above, and in a pseudo code:

a- *Determine tau and k value s, generate the tabu list with (time list records *2)record*
b- *Generate randomly the first generation and run to determine the gbest one as a solution sol.*
c- *First loop while not reaching repeat number=3:*

Second loop while not reaching last g=96:

1- *Generate a new generation which contains (time list records *2) lists, by increasing and decreasing the **changed number** (k) to records in the sol.*
2- *Run the new generation with SUMO to determine the best time list.*
3- *If (best < gbest): best is tabu and gbest = best, sol= best.*
4- *Else: if (best is not tabu): sol= best, else: sol =next list in generation.*

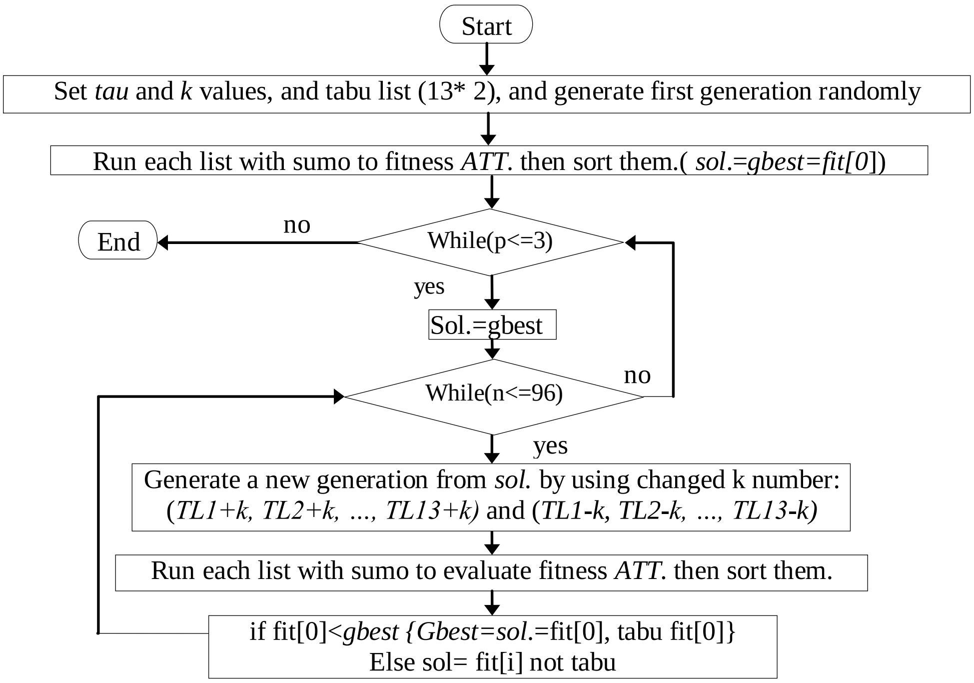


Figure 3.13: TS Type 4 algorithm flowchart

3.8.3.4. TS Type 4 Algorithm (TS 4): This type of TS algorithm is similar to TS 3 in all steps, but the reversing process for the best previous solution must be enforced at the beginning of the first loop of this algorithm to avoid worst solutions regions through the searching process.

The steps are shown in Fig.3.13 above and in a pseudo code:

*a- Determine tau and k value s, generate the tabu list with (time list records *2)record*

b- Generate randomly the first generation and run to determine the gbest one as a solution sol.

c- First loop while not reaching repeat number=3:

Sol= gbest

Second loop while not reaching last g=96:

*1- Generate new generation which contains (time list records *2) lists by increasing and decreasing the* ***changed number*** *(k) value to the records in the sol. list.*

2- Run the new generating with SUMO to determine the best time list.

3- If (best < gbest): best is tabu and gbest = best, sol= best.

4- Else: if (best is not tabu): sol= best, else: sol =next list in generation.

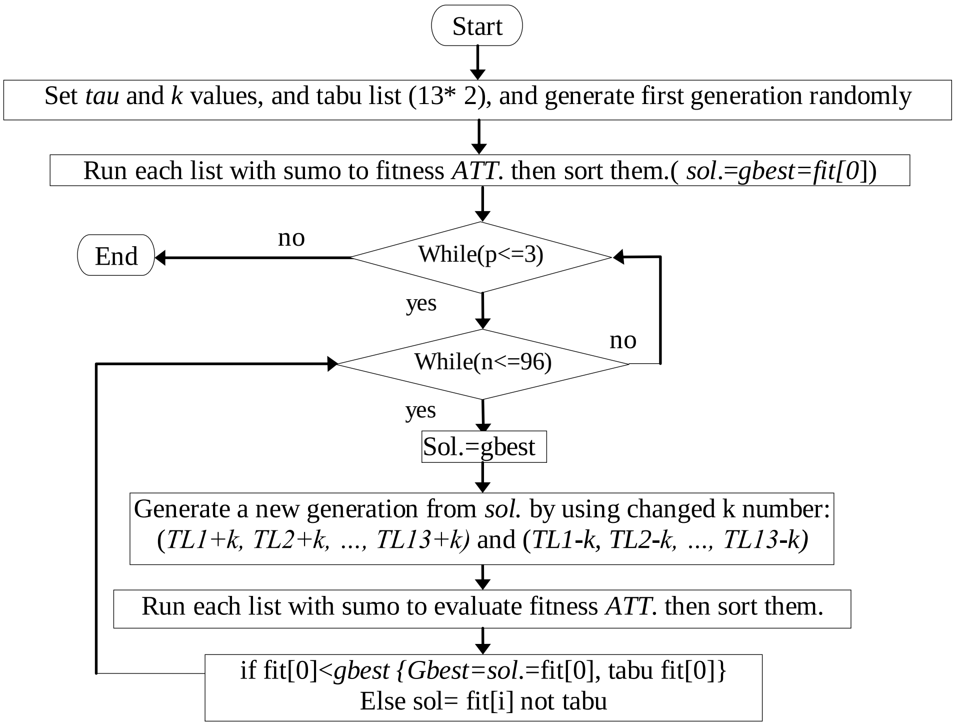


Figure 3.14: TS Type 5 algorithm flowchart

3.8.3.5. TS Type 5 Algorithm (TS 5): This type of TS algorithm is similar to TS 4 in all steps, but the reversing process for the best previous solution must be enforced at the beginning of the *second* loop of this algorithm to avoid worst solution regions through the searching process at the beginning of each new generation. The steps are illustrated in Fig.3.14 above and in a pseudo code:

*a- Determine tau and k value s, generate the tabu list with (time list records *2)record*
b- Generate randomly the first generation and run to determine the <u>gbest</u> one as a solution sol.
c- First loop while not reaching repeat number=3:
Second loop while not reaching last g=96:
Sol= gbest
*1- Generate new generation which contains (time list records *2) lists by increasing and decreasing the **changed number** (k) value to the records in the <u>sol.</u> list.*
2- Run the new generation with SUMO to determine the <u>best</u> time list.
3- If (<u>best</u> < <u>gbest</u>): best is tabu and gbest = best, sol= best.
4- Else: if (best is not tabu): sol= best, else: sol =next list in generation.

At the end of Tabu algorithm types section, in TS types 1 and 2, the *k* number added and dropped each record in the best solution list of the previous generation, to generate the next generation. This is a fixed number for all generations, like k={1, 2, 3,…, 20. The speed of arriving at the optimal solution was very slow because the steps of movement between the best solutions were very small and concentrated in some directions. For this reason, TS types 1 and 2 almost had the worst solutions or local optimal solution in solving both benchmark function and traffic light signals timing optimization problems, this can be seen in the results of experiments 7 and 8 (see pages 74-76).

However, TS types 3, 4 and 5 were improved in this research; *k* number added and dropped each record in the best solution list of the previous generation to generate the next generation. This is a changed number for each of many generations (see how *k* value changed in the experiments 7 and 8 in pages 61-62), it started from a big value until arrival at a small value like k= {45, 40, 30,…, 2, 1}. The speed of the arrival at the optimal solution was very fast because the steps of movement between the best solutions were very long and in all directions, and the search processes were in different region of the solution space. For this reason, TS types 3, 4 and 5 had more optimal solutions than the first two types in solving benchmark function. TS type 5 at *tau =10* almost had the nearest solution to the global optimal solution in solving traffic light signals timing optimization problems, as can be seen in the results of experiments 7 and 8 (see pages 74-76).

Chapter four

Methodology of the Study

The methodology of the research depends on using benchmark iterative approach with metaheuristic optimization techniques. This research is classified into both quantitative and experimental types.

Statement of the problem: Can we benefit from using a benchmark iterative approach to determine the best parameters' values in a short time for each metaheuristic optimization algorithms to be more efficient for optimizing the traffic light signals timing problem?

Research objectives: The aim of this study is to identify the best parameters' values for each metaheuristic optimization algorithm used in this research, thus making it more efficient to optimize traffic light signals timing problem.

Research population and sample: The target population of this study is Nablus City Road Network. The sample of the study, selected randomly, is the Road Network of the city center. It includes the intersection in front of Al-Watani Hospital to the eastern intersection of Nablus Municipality in the city center. The network contains 13 traffic light signals as discussed in chapter 3.

Research tools: The software tools (SUMO simulator, Python 2.7) are used; the hardware tools used were 12 computers with processor: Intel® Xeon® CPU E5603@ 1.600 GHz, Ram: 12.0GB and system type: 64 bit operating system, in addition to 40 computers with processor: Intel® core (tm) i7-3770 CPU 3.40 GHz (8 cpus), Ram: 8.0 GB and system type: 64 bit operating system All were in two computer labs at An-Najah National University.

Research methods: Methods used included a random algorithm and three metaheuristic optimization algorithms: three types of GA, PS, five types of TS (For details, see chapter three).

Research hypothesis: There is a relation between metaheuristic optimization algorithms' parameters' values and the efficiency (ex. finding a solution closer to the optimal solution) of the algorithms in finding the optimal or near optimal solution for traffic light signals timing optimization problem.

Research procedure: The research procedure depends on eight experiments:

1-Traffic Light Signals Timing Experiment 1 with GA: GA types 1, 2 and 3 and random algorithms were used to optimize traffic light signals timing problem. Fixed conditions were assumed for these algorithms in this experiment:

a- Mutation probability (*MP*) parameter changes at each different population, as *MP*=[10%, 20%,30%, 40%,50%,60%,70%,80%,90%]. Therefore, each algorithm of GA Type 1 and GA Type 2 were experimented with nine different populations.

b-Each population consisted of 50 generations, and each generation consisted of 30 chromosomes. That is, the number of iteration times for running SUMO simulator was 50*30=1500 iterations. The number was used in each experiment and for each algorithm.

c-This experiment was repeated 10 times for each algorithm of GA types 1, 2, and 20 times for GA Type 3 and random algorithm.

2-Benchmark Function Experiment 2 with GA: This represents the start of the benchmark iterative approach. This experiment is similar to Experiment 1 with the same algorithms and conditions used; the second one is with a benchmark problem to see/test the behavior of each algorithm and how algorithms' results are far from the optimal solution.

3-Benchmark Function Experiment 3 with GA and PS algorithms: In this experiment, the researcher tried to improve the results of the algorithms used in Experiment 2, by increasing the generation size and number for each GA type (1, 2 and 3), PS with parameters' values *w*=0.25, *cp*=1, *cg*=2 were assumed and random algorithms were experimented until arriving at the optimal solution for the benchmark function by GA Type 3 at *generation size* =50 and *generation number* =150. The experiment conditions are as follows:

a-The first condition in this experiment is similar to the first condition in Experiment 1.

b-Each population consists of 150 generations, and each generation consisted of 50 chromosomes. That is, the number of iteration time of benchmark function running is 150*50= 7500 iterations in each experiment and for each algorithm.

c-This experiment is repeated 10 times for GA types (1, 2), and 20 times for GA Type 3 and PS algorithm, and 110 times for random algorithm.

4-Traffic Light Signals Timing Experiment 4 with GA and PS algorithms: This experiment is similar to Experiment 3 with the same algorithms and conditions, but the second one was with SUMO simulator to optimize traffic light signals timing problem to see the optimal solution of each algorithm if improved at *generation size*= 50 and *generation number*= 150.

5-Benchmark Function Experiment 5 with PS Algorithm: In this experiment, improved the PS algorithms were experimented by testing all *w*, *cg* and *cp* parameters' values with the same conditions as Experiment 3. Therefore, to get the best values of the three parameters, *w*, *cp* and *cg*, which could get the optimal or near optimal results, this experiment is done by following a number of steps and conditions:

a-Each parameter has a changed value which starts from 0 value to 5 by adding 0.25 each time it was presupposed. Each parameter had 21 different values as {0, 0.25, 0.5,…, 5}, and the number of different experiments tested =21*21*21.

b- After conducting these experiments many times, the best values of the three parameters, when the optimal results<=10, were taken. This step is repeated two times.

c-After that, the best parameters' values, when the optimal results of PS <=10, were detected for the benchmark function problem.

6-Traffic Light Signals Timing Experiment 6 with PS algorithm: In this experiment, the best two values, for *w*, *cp* and *cg* parameters, were determined in the light of the results of Experiment 5. They were also tested in this experiment for traffic light signals timing problem, but the results were worse than the results in Experiment 4.The researcher reversed the parameters' values by increasing the *cg* value and decreasing the *cp* value. Many values of the three parameters were experimented for traffic light signals timing problem with SUMO simulation under the same conditions of Experiment 3, Experiment 4, and Experiment 5.

7-Benchmark Function Experiment 7 with TS Algorithm: In Tabu Search algorithm, the main challenges were to determine the best *tau* values and what the best fixed or changed *k* values are and which algorithm arrived at the optimal benchmark results. In this experiment, TS types 1, 2, 3, 4 and 5 algorithms were used to optimize the benchmark function. The fixed conditions and constraints were as follows:

a-For all algorithm types, *tau* values ={ 1,2,3,…,12} were tuned.

b-For each algorithm type, the first generation started with a random of 50 lists, and then ran all lists with a benchmark function. The lists were sorted are dependent on the results. At the end, the list which produced the minimum result was selected to continue as a solution to get new generation lists.

c-Each population consisted of 3 levels with 96 times, and each generation consisted of 26 lists. That is, the number of benchmark function running times should be equal to 50+3*96*26= 7538 running times in each experiment for TS types 1, 2, 3, 4 and 5 algorithms.

d- Fixed numbers, k= {0.1, 0.2, 0.3, …, 2.5}, were used to add and subtract from each record in the solution list to get new generation. After that, this experiment was repeated 10 times in Tabu algorithm Type 1 and Type 2 for each k number.

e- For TS algorithm's types 3, 4 and 5, changed number k was used in this experiment as follow: the iteration number (j=96) was divided into 12 parts as:

at $0<= j <5 \rightarrow k=5.2$, at $4< j <10 \rightarrow k =2.5$, at $9< j <15 \rightarrow k =2$, at $14< j <20 \rightarrow k =1.5$.

at $19< j <30 \rightarrow k =1.2$, at $29< j <40 \rightarrow k =1$, at $39< j <50 \rightarrow k =0.8$, at $49< j <60 \rightarrow k =0.7$.

at $59< j <70 \rightarrow k =0.5$, at $69< j <80 \rightarrow k =0.3$, at $79< j <90 \rightarrow k =0.2$, at $89< j <96 \rightarrow k =0.1$.

k changed from one part to another part (k was same for many generations, then changed), to add to and subtract k from each record in the solution list. This experiment was repeated 50 times by TS Types 3, 4 and 5 for each algorithm.

8-Traffic Light Signals Timing Experiment 8 with TS Algorithm: In this experiment, TS types 1,2,3,4 and 5 algorithms, were used as they were used in Experiment 7, but in traffic light signals timing problem with SUMO simulator, the fixed conditions and constraints were presupposed for these algorithms in this experiment as follow:

a-For TS algorithm types 1 and 2, at *tau* =6, and TS algorithm types 3, 4 and 5 at *tau* = { 2, 4, 6, 8, 10}, were used in each experiment.

b-This point presented the same step 3 in the previous experiment, but with traffic light signals timing problem and SUMO simulation.

c- The results of all fixed numbers and *tau* values used in Experiment 7 were far from the optimal solution so; these numbers could not be experimented in the SUMO simulator because the number of experiments needed long times to test them. Therefore, *tau* =6 was selected to be used in TS algorithm types 1 and 2, with fixed number k= 7. This experiment was repeated 10 times for Tabu algorithm types 1 and 2.

d-For TS algorithm's types 3, 4 and 5, the changed number k was used in this experiment as follow: the iteration number (j=96) was divided into 12 parts as:

at $0<= j<5 \rightarrow k=45$, at $4< j<10 \rightarrow k=30$, at $9< j<15 \rightarrow k=25$, at $14< j<20 \rightarrow k=20$.

at $19< j<30 \rightarrow k=17$, at $29< j<40 \rightarrow k=15$, at $39< j<50 \rightarrow k=12$, at $49< j<60 \rightarrow k=10$.

at $59< j<70 \rightarrow k=7$, at $69< j<80 \rightarrow k=5$, at $79< j<85 \rightarrow k=3$, at $84< j<90 \rightarrow k=2$.

and at $89< j<96 \rightarrow k=1$.

k changed from one part to another part (k is the same for many generations, then changed), to add to and subtract k from each record in the solution list. This experiment was repeated 10 times for each of TS algorithm Types 3, 4 and 5.

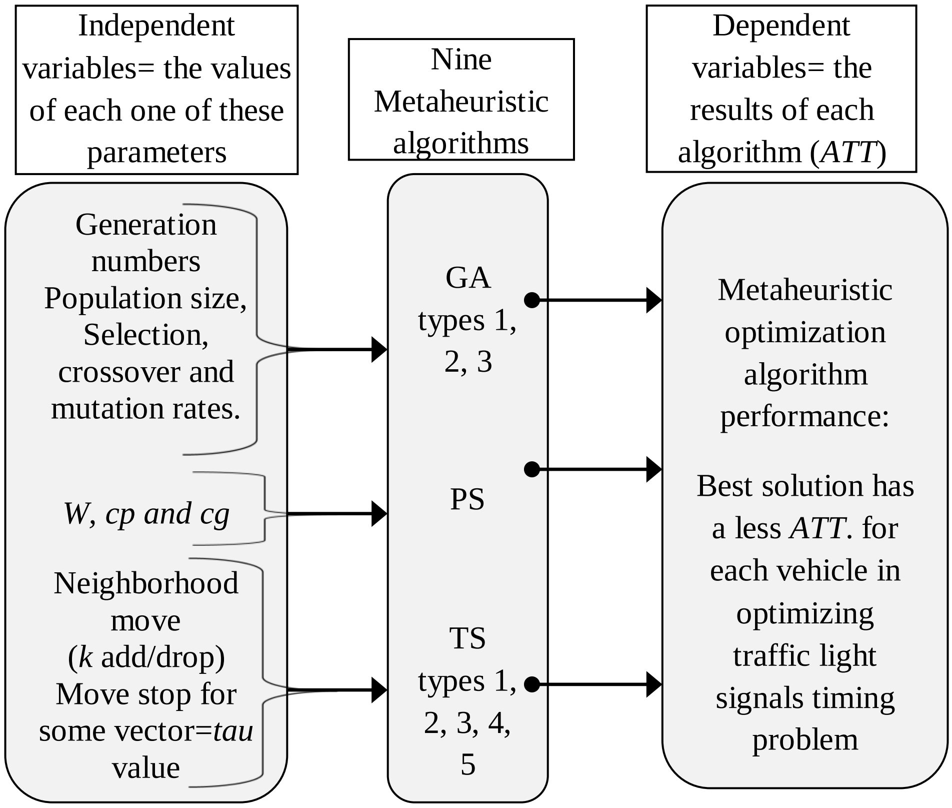


Figure 4.1: Research Design

Research design: (see Figure 4.1 above)

Research results analysis: Using T test, significant intervals, min., max. and the average, the algorithms' results were analyzed and tested in chapters five and six to ensure the benefits from the benchmark iterative approach. These results were validated by comparing the algorithms' results after and before using the benchmark iterative approach with the Webster and HCM methods and SYNCHRO simulation.

Chapter five

Results and Discussion of Experiments

This chapter presents and discusses chronologically the results of the experiments. Each table in Appendix 1 represents one experiment results. There are many differences between algorithms' results in each experiment as shown in the tables. The comparison process between algorithms' results depended on comparing the average, as well as maximum and minimum results.

* TLSTP: Traffic Light Signals Timing Problem

5.1 Results of Experiment 1: [Comparison between GA types 1, 2 and 3 in TLSTP]

Firstly, from the results of this experiment, the researcher found that the probability of mutation operation had an effect on the optimal solution which arrived at by both GA Types 1 and 2, and the best average for the optimal solution when *MP* = 70%, for both algorithms as shown in Fig.5.1 below and Table 5.1 (see Appendix 1).

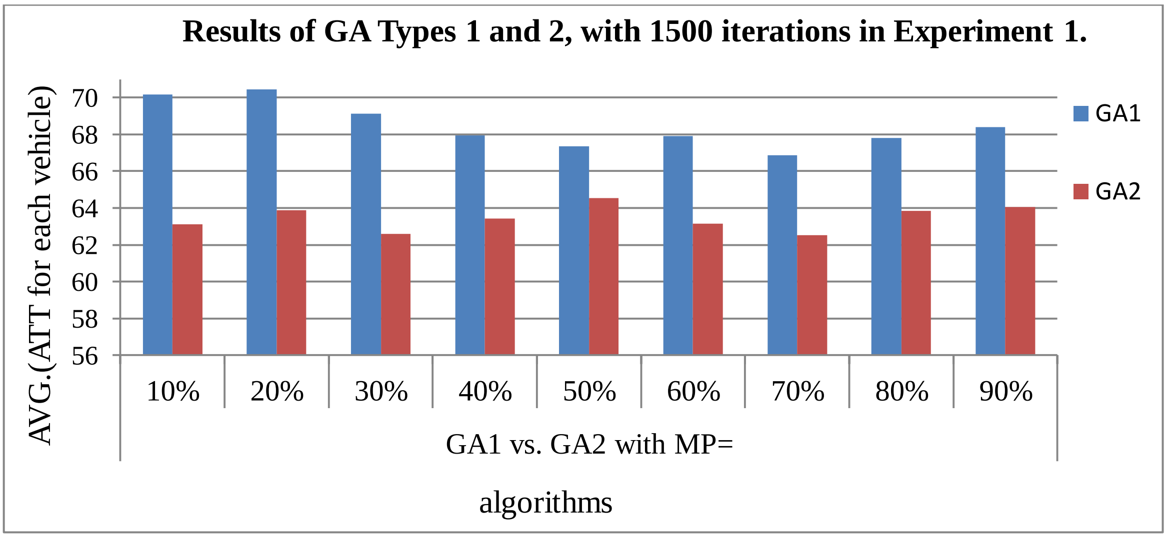


Figure 5.1: Results of GA 1 vs. Results GA 2 in Experiment 1

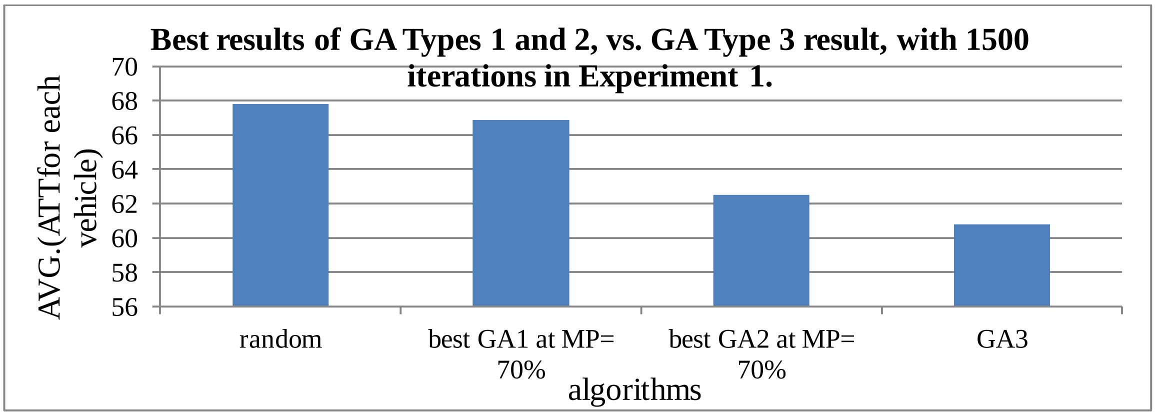


Figure 5.2: Summary of Experiment 1 results

Secondly, the results of this experiment show that the four algorithms were not equal and the best one was GA Type 3. According to the average results of each algorithm, GA Type 3 had the minimum average results and the minimum result, and the worst one was GA Type 1. Its average results were near the random algorithm results as shown in Fig.5.2 above and Table 5.1 (see Appendix 1).

The preferable algorithm results were those of GA Type 3 because the selection process transferred the best half of chromosomes from the previous generation to the next generation without any random chromosome as in GA types 1 and 2. Fig.5.2 above summarizes the results in Table 5.1 (see Appendix 1).

Because of the differences in GA types' results, the main hypothesis was accepted for the first time, and the selection, crossover and mutation rate were found to have an effect on the GA performance, but at the end of this experiment, we can ask these questions:

Is the GA Type 3 the best algorithm to optimize traffic light signals problem?

Can these algorithms become more improvable and more efficient in optimizing this problem?

Is the minimum result of the GA Type 3 the optimal solution for the traffic light signals timing optimization problem in the research case study road network?

To answer these questions, Experiment 2 with the benchmark function was done.

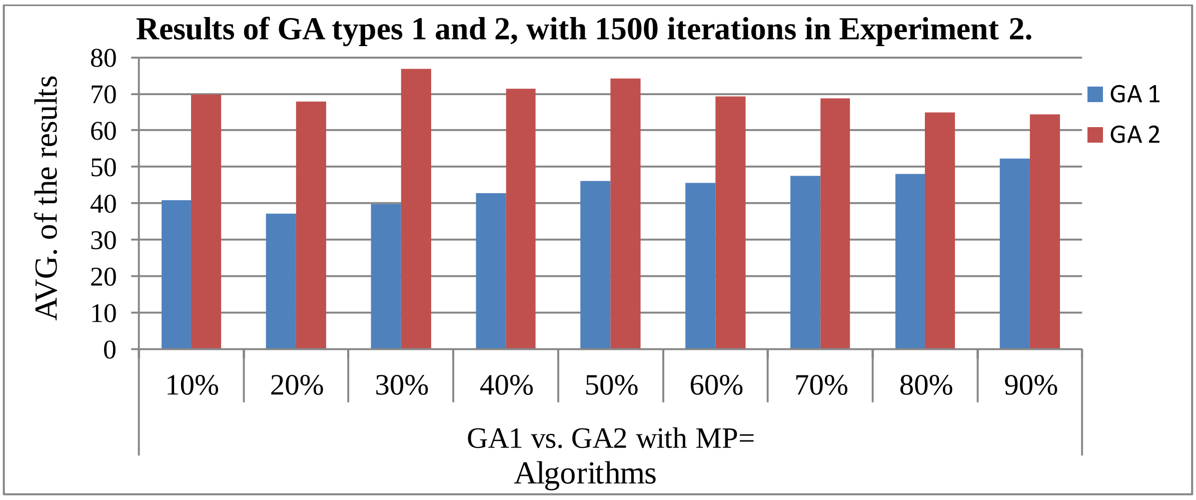


Figure 5.3: Results of GA 1 vs. Results GA 2 in Experiment 2

5.2 Results of Experiment 2: [comparing GA types 1, 2 and 3 in benchmark]

As it was illustrated in chapter three, the optimal result of the benchmark function was known=0. From the results of this experiment, the researcher found that the probability of mutation operation also had an effect on the optimal solution which was arrived at by both GA Types 1 and 2, and the best average for the optimal solution when *MP* = 20% for GA Type 1 and *MP* = 90% for GA Type 2, as shown in Fig.5.3 above and Table 5.2 (see Appendix 1).

The results of this experiment show that GA Type 3 was the most suitable one, because the average results were less than the average results of the other algorithms (*AVG.* = 16.1). The best GA Type 1 results (*AVG.* = 37) were better than the best GA Type 2 results (*AVG.* = 64), but they were far from optimal result *y*= 0. The results of the three algorithms, GA Type 1, GA Type 2 and GA Type 3, were better than the random algorithm as shown in Fig.5.4 on the next page which summarizes the results in Table 5.2 (see Appendix 1).

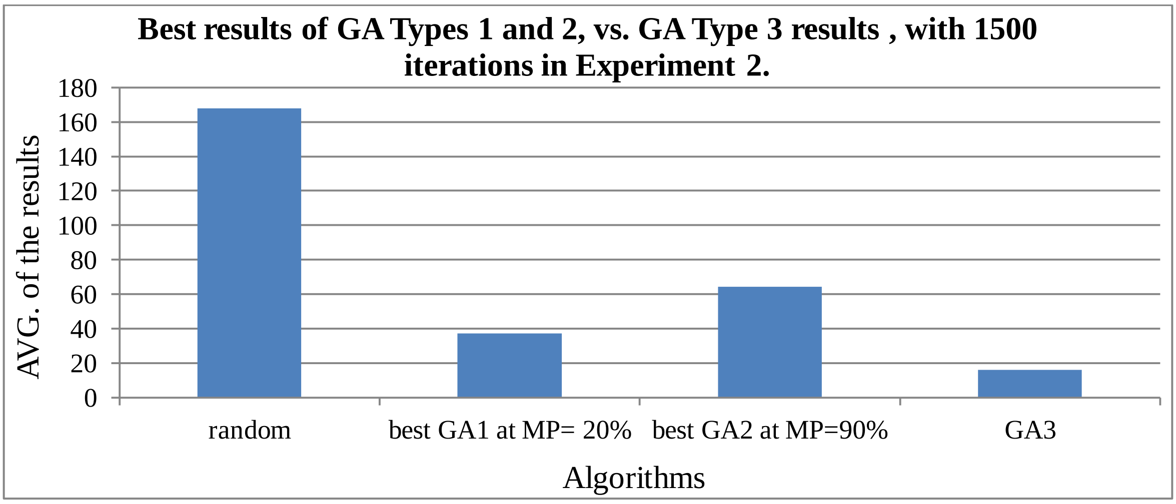


Figure 5.4: Summary of Experiment 2 results

The main conclusion from this experiment is that the results of the algorithms were still far from the optimal result of the benchmark function y=0, but the best optimization algorithm for the benchmark function must arrive at the optimal value y=0 or $y \cong 0$ value. The researcher tried to improve the performance of GA types 1, 2 and 3 and PS algorithms by finding a process for the optimal result of the benchmark function and by increasing the number of the execution times for the benchmark evaluation in the search process. This could be through increasing population size and the number of generations in Experiment 3 to see if the results improved or unchanged.

5.3 Results of Experiment 3: [Tuning GA Types 1, 2 and 3 and PS in benchmark]

Through increase of the process of the execution times parameter, such as population size and number of generations, the results of the optimization algorithms used in Experiment 2 improved in this experiment. The best average results of GA Type 1 =33, at *MP*= 20%, and the best average results of GA Type 2 =36, at *MP*= 30% are shown in Fig.5.5, next page and Table 5.3 (see Appendix 1).

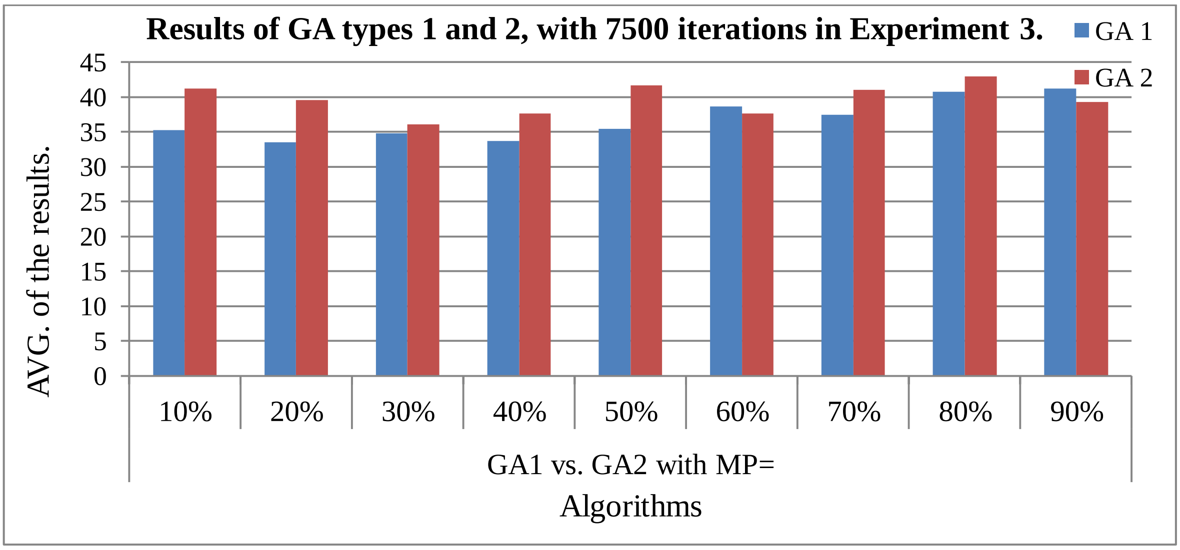


Figure 5.5: Results of GA 1 vs. Results GA 2 in Experiment 3

The best improvement was in GA Type 3 in *generation size*=50 and *generation number*=150. Its results were the closest to the optimal of the benchmark result *y*=0.The researcher stabilized this pop. (Size) and G number for all algorithms. The worst algorithm was PS algorithm because its results were the farthest from the optimal of the benchmark, as shown in Fig.5.6 below which summarizes the results in Table 5.3 (see Appendix 1).

From the results of this experiment, one can conclude that the execution times or the number of times parameter value had a strong effect on the metaheuristic optimization algorithms' performance in finding the optimal or near optimal solution.

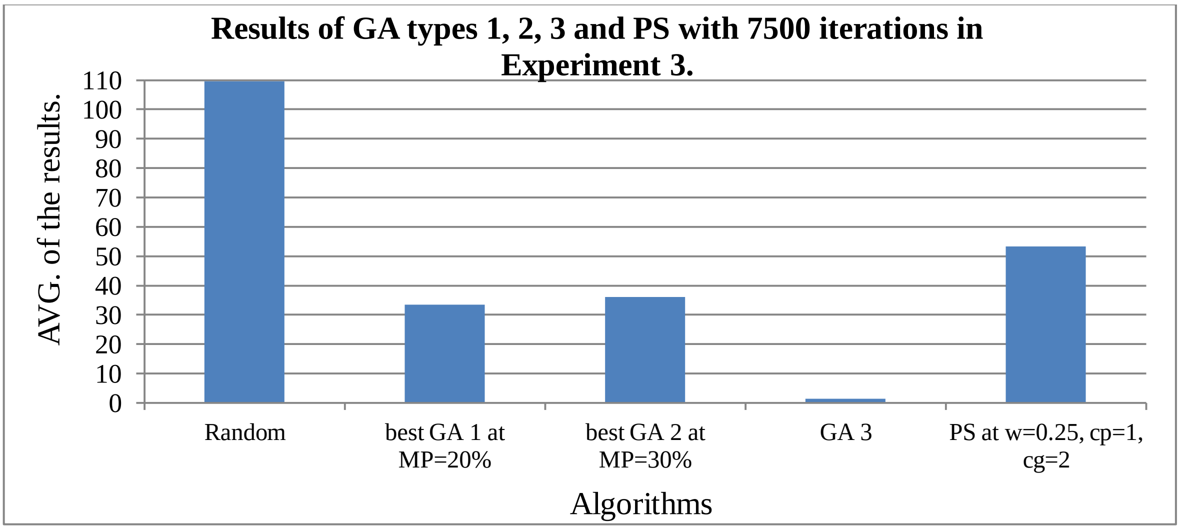


Figure 5.6: Summary of Experiment 3 results.

When we increased the population size and the generation numbers, GA types 1, 2 and 3 algorithms' results improved with different percentages. The main reason for the differences between the results of the algorithms was that the best algorithm had a high power to search in all regions of the search space. However, the worst algorithm had just a central search in some regions which had always local minima.

Then the same optimization algorithms (GA1, GA2, GA3) and PS algorithm and random algorithm, were used in this experiment under the same conditions of 50 chromosomes and 150 generations were used to optimize the traffic light signals timing problem with SUMO simulator in Experiment 4 to see if the results improved in comparison with the results in Experiment 1.

5.4 Results of Experiment 4: [comparing tuned GA types 1, 2 and 3, and PS in TLSTP]

The results of this experiment showed that there were some improvements on the results of the metaheuristic optimization algorithms GA types 1, 2 and 3 which were used in Experiment 1. That was because the average results and the minimum result of each algorithm in this experiment were less than the average results and the minimum result of each algorithm in Experiment 1. After comparison the average results of the *ATT* for each vehicle which were produced from GA Types 1 and 2, the best *ATT* results of the GA Type1 = 58s, at *MP* = 40%, and the best *ATT* results of the GA Type 2 = 60s, at *MP* = 10%, as shown in Fig.5.7 on the next page and Table 5.4 (see Appendix 1).

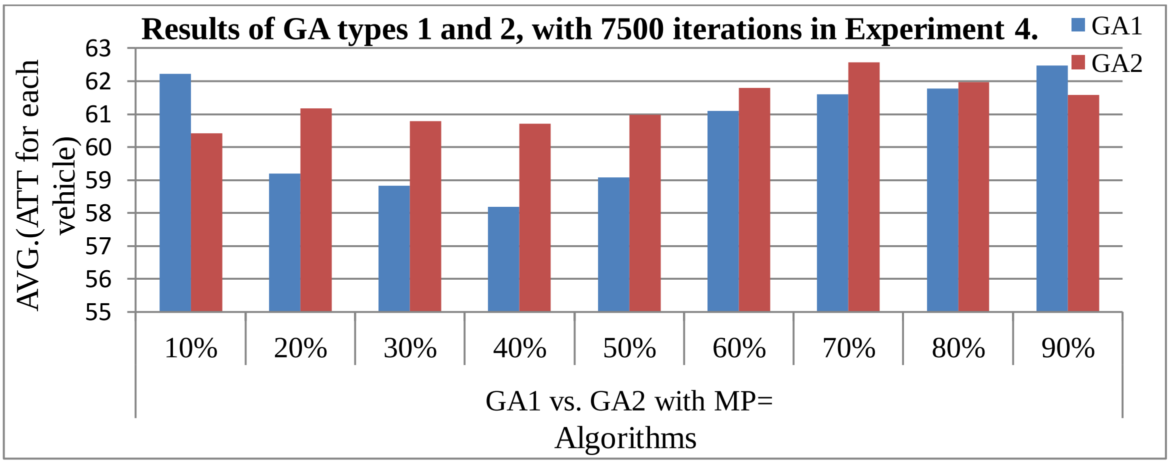


Figure 5.7: Results of GA 1 vs. Results GA 2 in Experiment 4

The best algorithm in this experiment was GA Type 3 because the average results of the average travel time for each vehicle was the minimum (*AVG*.(*ATT*) = 55.9s), and the minimum result of the average travel time for each vehicle *min*.(*ATT*)=53.4s, as shown in Fig.5.8 below and Table 5.4 (see Appendix 1). These improvements were right logically because the number of search times increased from 1,500 times in Experiment 1 to 7,500 times in this experiment. The main observation, in this context, was when the GA Type 3 was saved for the preferable results in the four previous experiments. The main research hypothesis was accepted because the number of times parameters' values (generation size and generation number) had a strong effect on the metaheuristic optimization algorithm performance in the optimizing process of the traffic light signals timing problem.

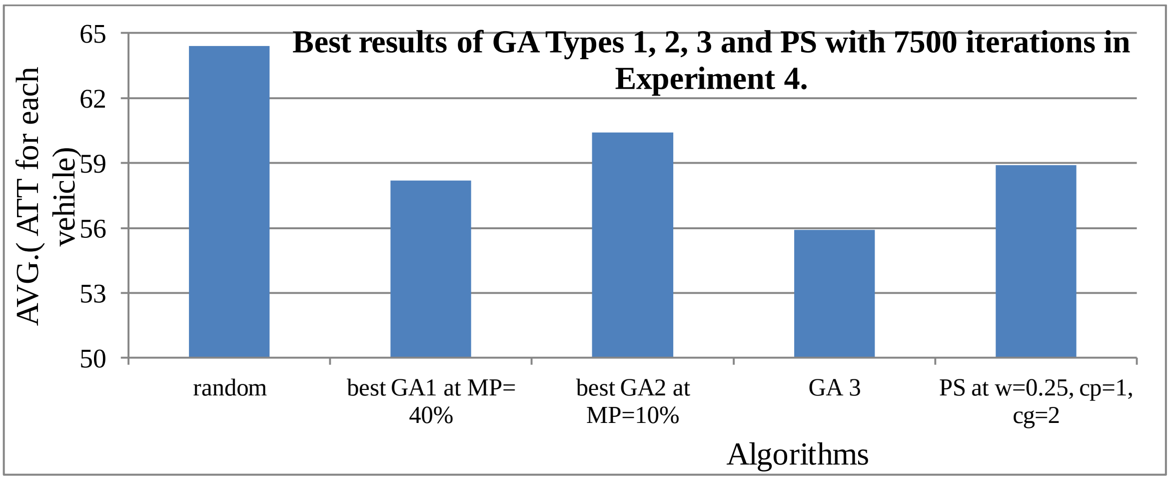


Figure 5.8: Summary of Experiment 4 results.

The results of *AVG.*(results)=53.3 of the PS algorithm were far from the optimal benchmark function value y=0 in Experiment 3. In addition, the PS algorithm results were not the best results in Experiment 4. Experiment 5 was done to improve the performance of PS algorithm to get the optimal or near optimal solution for the benchmark function by finding a process for the best parameters' values (*w, cp, cg*).

5.5 Results of Experiment 5: [tuning PS by using benchmark]

- First, after completion of this experiment with the parameter values:

  w = {0, 0.25, 0.5 , 0.75, 1.25, … , 5}, and *cp* ={0, 0.25, 0.5 , 0.75, 1.25, … , 5}, and *cg*= {0, 0.25, 0.5 , 0.75, 1.25, … , 5} the total number of the experiments was 21*21*21=9261 as the number of the probabilities for the *w* ,*cp* and *cg* values. These three values were done twice. The results were found to be very good, and some experiments results were almost close to the optimal solution of the benchmark. Therefore, the experiments whose results were ="0" or <= "10" were taken. The three parameters' values of each experiment results <=10 were when: *w*= [0, 0.25, 0.5], *cg*= [0.75, 1, 1.25, 1.5, 1.75, 2, 2.25, 2.5, 2.75], and *cp*= [1, 1.25, 1.5, 1.75, 2, 2.25, 2.5, 2.75, 3, 3.25].

- Second, each experiment was done two times for each probability for the best values of *w, cg* and *cp* produced in first step. The best results were less than 10 when:

  w= [0, 0.25], *cg*= [0.75, 1, 1.25, 1.5, 1.75, 2], and *cp*= [1.75, 2, 2.25, 2.5, 2.75, 3, 3.25].

- Third, each experiment was done forty times for each probability of the three parameters' values (*w, cg, cp*), which produced the best results in the second step. The best experiments' produced results were less than 10 when :

{*w*, *cg*, *cp*}= {0, 0.75, 2.25} with 17 times less than 10, {0, 0.75, 2.5} with 20 times less than 10, {0, 0.75, 2.75} with 15 times less than 10, { 0, 0.75, 3} with 19 times less than 10, and {0.25, 0.75, 2.75} with 15 times less than 10.

- Fourth, the best three parameters' values, which produced the best results in the third step, were tested 20 times for each three parameters' values. The results are shown in Table 5.5 (see Appendix 1).

As 5.5 (see Appendix 1) shows, the best two results were when the number of times their results were less than 10 more than half of the total number of times. This was done in probabilities 2 and 3 as shown in Table 5.5 (see Appendix 1).Therefore, the best results were when: {*w*, *cg*, *cp*} = { 0, 0.75, 2.5} and { 0, 0.75, 2.75}. The best values of the three parameters (*w*, *cg*, *cp*), which produced the best two results in the benchmark function, were tested in solving traffic light signals timing optimization problem with SUMO simulator in Experiment 6.

5.6 Results of Experiment 6: [using tuned PS in TLSTP]

After the experiment of the best two values of the three parameters {*w*, *cg*, *cp*} = {0, 0.75, 2.5} and {0, 0.75, 2.75} for PS algorithm, which produced the best results in the previous experiment, in optimizing the traffic light signals timing problem with SUMO simulator in this experiment, the results were found to be not good as shown in Fig.5.9, next page, and Table 5.6 (see Appendix 1). They were worse than the results of PS algorithm in Experiment 4 at parameters' values {*w*, *cg*, *cp*} = {0.25, 2, 1}.

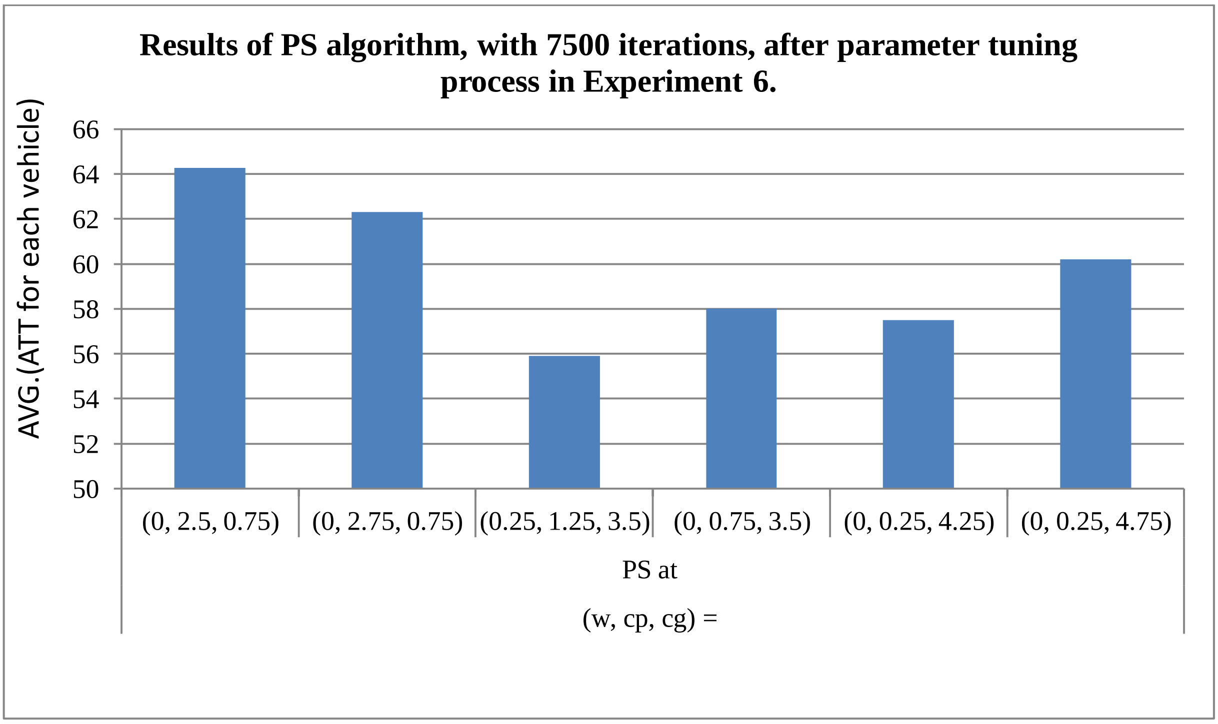


Figure 5.9: Summary of Experiment 6 results.

Therefore, the parameters' *cg* and *cp* values {0.75, 2.5} and {0.75, 2.75} were reversed by increasing the cg value and decreasing the cp value, and the parameters' values tried were *w*, *cg*, and *cp*= {0, 4.75, 0.25}, {0, 4.25, 0.25}, {0, 3.5, 0.75}, {0.25, 3.5, 1.25} as shown in Fig.5.9, above and Table 5.6 (see Appendix 1). The results were produced were good, and the best results were at the parameter's values *w*, *cg* and *cp* = {0.25, 3.5, 1.25} as shown in Fig.5.9, above and Table 5.6 (see Appendix 1).

Concerning the average results of PS algorithm in this experiment *AVG*.(*ATT*) = 55.9s and the minimum results *Min*.(*ATT*)=53.4s when the parameters' values were *w*, *cg*, and *cp*={0.25, 3.5, 1.25}, it was found that they were better than the average results of the same algorithm in Experiment 4 *AVG*.(*ATT*)=58.9s and the minimum results *Min*.(*ATT*)= 54.9s were when the parameter's values were *w*, *cg* and *cp* and = {0.25, 2, 1}. From these results, we conclude that the values of the three parameters, *w*, *cp* and *cg*, had a direct effect on the efficiency of PS algorithm. Therefore, the main research hypothesis was accepted again.

5.7 Results of Experiment 7: [tuning TS by using benchmark]

This experiment was done by using TS algorithm types 1, 2, 3, 4, and 5, with benchmark function, to detect the best parameters' values, as *k* number and *tau* value, and to get the optimal or near optimal results for the benchmark function. The main challenge was finding the best values of these parameters. The TS algorithm has a high performance to get the optimal or near optimal results for the benchmark function.

The results of the standard TS algorithms, such as TS Type 1 and Type 2 algorithms' were not good as shown in two tables: Table 5.7 and Table 5.8 (see Appendix 1). The best results of all *k* fixed values = {0.1, 0.2, 0.3, …, 2.5}and *tau* values ={1, 2, 3, …, 12}, tested, were greater than 20. These results were farther than the optimal of the benchmark function y= 0 because the search processes maybe central in some regions which have a local minima.

In contrast, the results of the new algorithms' TS types 3, 4 and 5 were more efficient than the results of the standard algorithms TS types 1 and 2 as shown in Table 5.9 (see Appendix 1).The new four algorithms' types produced the optimal benchmark function value y=0, because the *k* number changed as it was illustrated in the experiment conditions in chapter four, this allowed to search processes in all the regions of the solution space, and this was almost the main reason for the ease of obtaining the global minimum results using these algorithms' types. The best type was TS Type 5. All its results arrived at the optimal value y=0, and the percentage of the results = 0 was 100% as shown in Table 5.9 (see Appendix 1).

5.8 Results of Experiment 8: [using tuned TS in TLSTP]

After completion of the previous experiment, and because the results were promising especially in using new algorithms TS type 3, 4 and 5 algorithms, these algorithms were experimented in optimizing traffic light signals timing problem with SUMO simulator in this experiment. The big number of probabilities input fixed parameters' values was k= {0.1, 0.2, 0.3, …, 2.5} and tau= {1, 2, 3, …, 12} were experimented with in the previous experiment. This number of probabilities could not be done to optimize traffic light signals timing problem with SUMO simulator in this experiment. They may need several months to finish these probabilities of parameters' values.

One parameter's value for each k and tau as k= 7 and tau= 6 was tested by each standard algorithm TS types 1 and 2 in this experiment. Both results were bad because they were close to the full random algorithm results or perhaps they were worse than the full random algorithm results as shown in Table 5.10 (see Appendix 1).

When k number changed, and five values of tau parameter ={2, 4, 6, 8, 10} were tested by new algorithms TS types 3, 4 and 5 in this experiment, the best algorithm was found to be TS type 5 algorithm because all the averages of the results at all Tau values were less than 58.5s, and the minimum average results of this algorithm was $AVG.(ATT)$= 56.2s and the minimum result of this algorithm was $Min.(ATT)$= 52.5s at tau= 10. The worst algorithm was TS Type 3 algorithm, because all the averages of the results at all tau values were more than 61.5s. The main reason for these differences, between the results of the three TS algorithm types, was the start of input list in each stage of the algorithm. TS Type 3 algorithm started each time of the two stages from a new solution which is unknown. This is the reason for its bad results. But TS Type 4 algorithm started each time in the first stage from the best solution which the algorithm arrived at. This is the reason why its results were better than those of TS Type 3. At the end, TS Type 5 algorithm started each time of the second stage from the best solution which the algorithm arrived at.

This is the reason why its results were the best of the three algorithms as Table 5.10 shows (see Appendix 1). At the end of this experiment, *tau* and *k* values were found to have a strong effect on the performance of TS algorithm. Thus, the main hypothesis was accepted again.

5.9 Best and Worst Algorithms Behavior and Decisions

All traffic light signals timing experiments with SUMO simulator were divided into two parts. The first part was the number of execution times =1,500 time as in Experiment 1, and the second part was the number of execution times =7,500 time as in three experiments (4, 6 and 8). Each algorithm was used in any part had a better result when the result was a minimum and a worse result when the result was maximum.

In the first part, the best behavior cases for each algorithm, used in Experiment 1, are shown in Fig.5.10 below. It is clear in this figure that the behavior of each algorithm, according to the generation numbers, and the algorithm GA Type 3, was the best one. The algorithm GA Type 1 was the closest to the random algorithm.

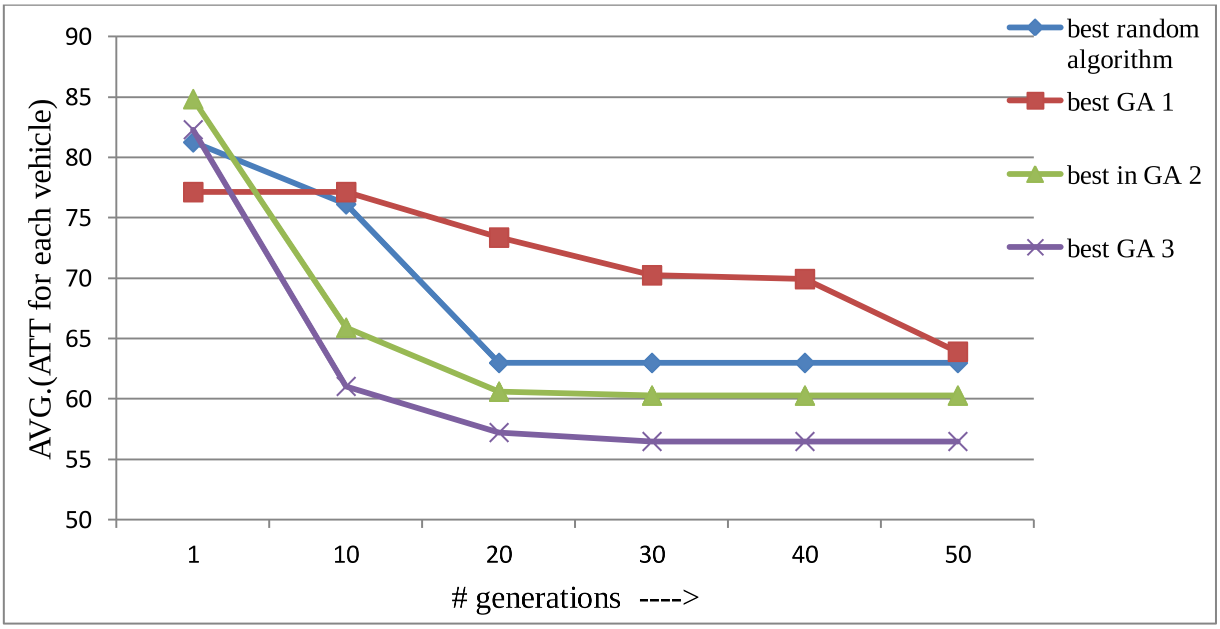


Figure 5.10: Best case of algorithms' results of Experiment 1

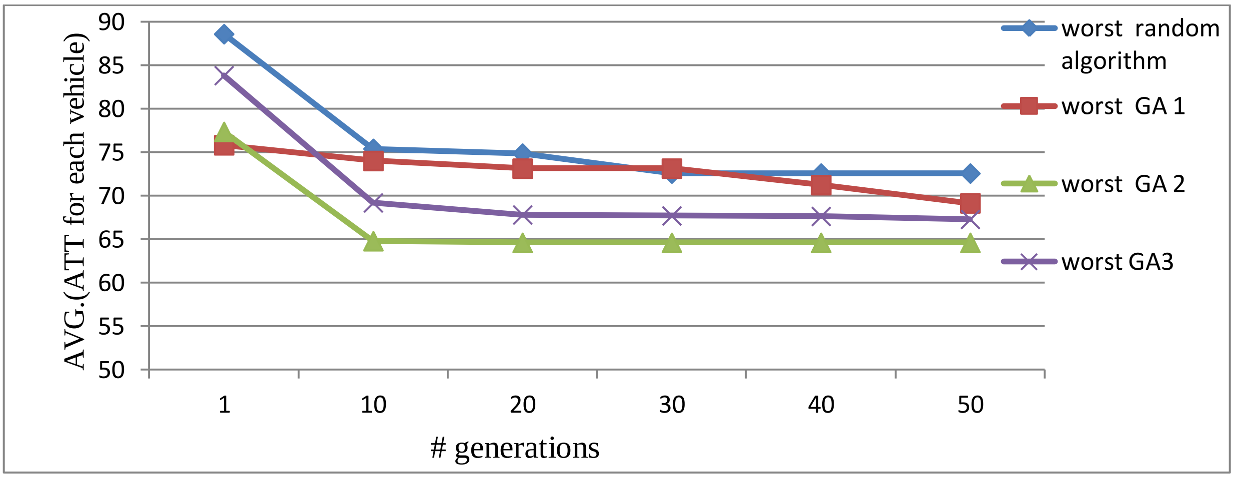


Figure 5.11: Worst case of the algorithms' results of Experiments 1.

The worst cases for each algorithm used in Experiment 1, are shown in Fig.5.11 above. This figure shows the behavior of each algorithm with the generation numbers.

In the second part, the benefits of using the benchmark iterative approach are illustrated. The algorithms' results improved in three experiments (4, 6 and 8) according to the results of Experiment1 by determining the best parameters' values for each algorithm. The best cases for the algorithms used in three experiments (4, 6 and 8) are shown in Fig.5.12 below. It is crystal clear in this figure that the behavior of each algorithm with the generation numbers and the algorithm TS type 5 was the best one, and the random algorithm was the worst one.

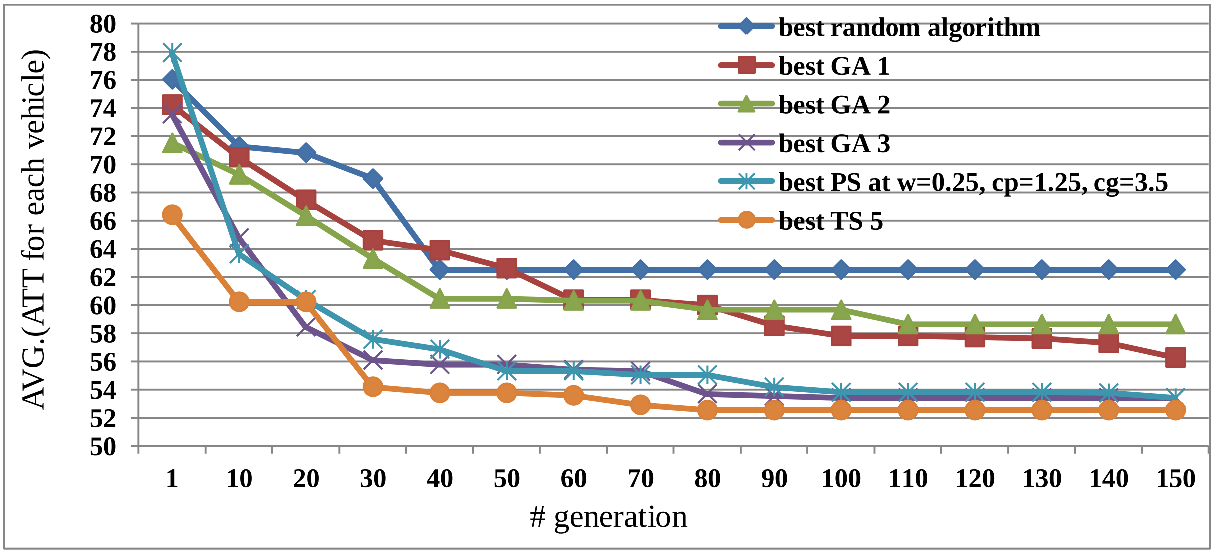


Figure 5.12: Best case of the algorithms' results of Experiments 4, 6 and 8

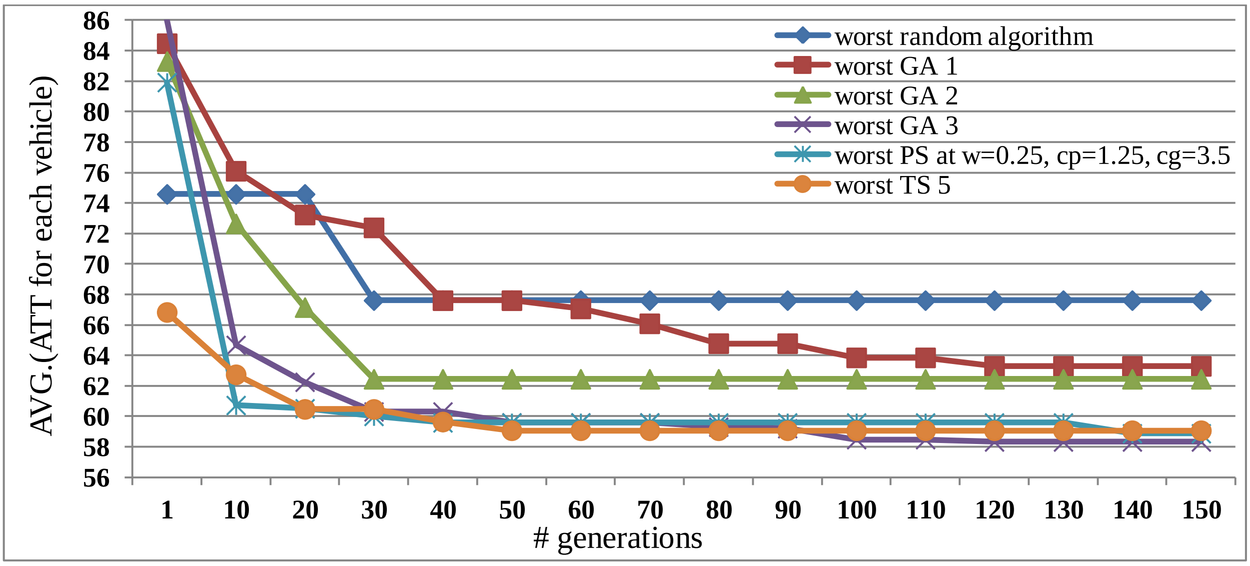


Figure 5.13: Worst case of the algorithms' results of Experiments 4, 6 and 8

The worst case for the algorithms used in the three experiments (4, 6 and 8) is shown in Fig.5.13 above. This figure shows the behavior of each algorithm according to the generation numbers.

At the end of this chapter, and by looking at the results in this chapter, the main conclusion is that the benchmark iterative approach could help in the improvement process of the metaheuristic optimization algorithms by determining the best parameters' values of each algorithm in optimizing traffic light signals timing optimization problem. The most suitable algorithms for optimizing the traffic light signals timing problem were GA Type 3, PS algorithm, at parameters' values w =0.25, cp =1.25 and cg =3.5, and TS type 5 algorithm, at parameters' values tau =10, with 7,500 execution times. This can be inferred by looking at the results of the following Tables: 5.4, 5.6 and 5.8 (see Appendix 1). They have the minimum average of *ATT* for each vehicle as *AVG.*(*ATT*) = 55.9s, for both algorithms of GA Type 3 and PS at w =0.25, cp=1.25 and cg =3.5, and the *AVG.*(*ATT*) =56.2s for TS Type 5 new algorithm at tau=10. They have the minimum *ATT* for each vehicle as *min.*(*ATT*)= 53.4s for both algorithms GA Type 3 and PS at w=0.25, cp=1.25 and cg=3.5 and the *min.*(*ATT*)=52.5s for TS Type 5 algorithm at tau=10.

However, to check this conclusion and the benefit of the benchmark iterative approach, the next chapter compares the results of the metaheuristic optimization algorithms used in this research before and after using benchmark iterative approach with the results of the mathematical models for optimizing the traffic light signals timing optimization problem, Webster and HCM methods, and the results of SYNCHRO simulator. The metaheuristic optimization algorithms' results were validated before using benchmark iterative approach and after using this approach. By using T test and confidence intervals (with 95% confidence), the researcher presents some reliability of the best algorithms' results.

Chapter six

Results Validity and Reliability

To validate the metaheuristic optimization algorithms' results, obtained from optimizing the traffic light signals timing problem in this research, the researcher compared these results, (Tables 5.1, 5.4, 5.6 and 5.8, Appendix 1) with the results of the common and traditional methods analytically like Webster and HCM and SYNCHRO simulation. These were used to determine the optimal timing for each traffic light signal in the road network of the case study (Fig.3.1) to minimize the average travel time for each vehicle (*ATT*). The results for each optimal time list from Webster and HCM methods and SYNCHRO simulation are shown in Table 6.1 below.

Table 6.1: Traffic light signals timing of Webster and HCM methods and SYNCHRO simulation and *ATT* results for each vehicle

HCM optimal timing for the traffic light signals at saturated lanes 1600 vph														
Traffic light signals time		***TL_0* phase 1**	***TL_1* phase 1**	***TL_2* phase 1**	***$TL_{3\ \&\ 4}$* phase 1**	**phase 2**	***$TL_{5\ \&\ 6}$* phase 1**	**phase 2**	***$TL_{7\ \&\ 8\ \&\ 9}$* phase 1 & 3**	**phase 2**	***TL_{10}* phase 1**	***$TL_{11\&12}$* phase 1**	**phase 2**	***ATT* for each vehicle**
Yellow		3	3	3	3	3	3	3	3	3	3	3	3	
at (v/c)=100%	green	3	6	4	6	2	13	5	6	2	3	6	3	153s
	red				2	6	5	13	2	6		3	6	
at (v/c)=90%	green	4	9	5	8	2	21	7	9	2	5	8	5	149s
	red				2	8	7	21	2	9		5	8	
at (v/c)=75%	green	7	20	9	14	4	67	21	16	3	9	15	9	178s
	red				4	14	21	67	3	16		9	15	
Webster optimal timing for the traffic light signals at saturated lanes 1600 vph														
Traffic light signals time		***TL_0* phase 1**	***TL_1* phase 1**	***TL_2* phase 1**	***$TL_{3\ \&\ 4}$* Phase 1**	**Phase 2**	***$TL_{5\ \&\ 6}$* Phase 1**	**Phase 2**	***$TL_{7\ \&\ 8\ \&\ 9}$* phase 1&3**	**phase 2**	***$TL_{10\&\ 11\ \&\ 12}$* phase 1**	**phase 2**	**phase 3**	***ATT* for each vehicle**
Yellow		3	3	3	3	3	3	3	3	3	3	3	3	
Green		13	23	23	19	3	41	11	29	3	22	22	3	202s
Red					3	19	11	41	3	29		3	22	
Synchro. optimal timing for the traffic light signals from Nablus municipality														
Traffic light signals time		***TL_0* phase 1**	***TL_1* phase 1**	***TL_2* phase 1**	***$TL_{3\ \&\ 4}$* Phase 1**	**Phase 2**	***$TL_{5\ \&\ 6}$* Phase 1**	**Phase 2**	***$TL_{7\ \&\ 8\ \&\ 9}$* phase 1&3**	**phase 2**	***$TL_{10\&\ 11\ \&\ 12}$* phase 1**	**phase 1**	**phase 1**	***ATT* for each vehicle**
Yellow		3	3	3	3	3	3	3	3	3	3	3	3	
Green		69	69	69	27	70	69	28	67	30	30	30	67	92s
Red					70	27	28	69	30	67				

All metaheuristic optimization algorithms' results, and random algorithm results, obtained before using the benchmark iterative approach or before tuning the algorithms' parameters' values from the experiments on the traffic light signals timing problem, using SUMO simulator, were found to be more suitable to optimize the traffic light signals timing problem than the results of the traditional methods. Metaheuristic optimization and random algorithms results of *ATT*. for each vehicle were in the range [*min*. =56.5s, *max*. =82s]. These were less than the results of *ATT*. for each vehicle produced from the analytical methods of Webster and HCM and SYNCHRO simulation. They were in the range [*min*.= 92s, *max*.=202s] as Table 6.1 shows. The main reasons of this conclusion are the following:

- ❖ Traditional methods of Webster, HCM and SYNCRO simulation computed the optimal timing for each isolated intersection, and the main factor here is the flow of vehicles on the lanes. However, this mathematical way may not be perfect especially when the flow of vehicles is small or huge. The timing phases might be illogical here as green time=2s or 250s.

- ❖ Metaheuristic optimization algorithms and random algorithm were used in this research to optimize the traffic light signals timing problem in a global way and actual experimental ways through trying 1,500 or 7,500 time lists in each algorithm.

After using the benchmark iterative approach and tuning process for the metaheuristic algorithms' parameters' values, all algorithms' results improved. Table 6.2 below shows that the benchmark iterative approach had an effect on the improvement of results of each algorithm used in Experiments 4, 6, 8. The importance of using the benchmark iterative approach was demonstrated because the metaheuristic algorithms' results, after using benchmark iterative approach and tuning process, were better than the

Table 6.2: Comparison between metaheuristic algorithms' results before and after using benchmark iterative approach in traffic light signals timing optimization problem

Algorithms' results (ATT.) for each vehicle before benchmark iterative approach and tuning process.					
algorithm	**Parameters values (supposed)**	**Iterations**	**Avg.**	**Min.**	**Max.**
random		1500	68s	63s	72.5s
GA1	*g size*=30, *#generations*=50	1500	68s	63s	73s
GA2	*g size*=30, *#generations*=50	1500	63s	58s	69s
GA3	*g size*=30, *#generations*=50	1500	61s	56.5s	67.5s
PS	*w*=0.25, *cp*=1, *cg*=2	7500	59s	55s	71s
TS(1 and 2)	*Tau*= 6, *k*= 7	7500	72.5s	64.5s	82.5s
Algorithms' results (ATT.) for each vehicle after benchmark iterative approach and tuning process.					
algorithm	**Parameters values (tuned)**	**iterations**	**Avg.**	**Min.**	**Max.**
random		7500	64.5s	62.5s	67.5s
GA1	*g size*=50, *#generations*=150	7500	60.5	56.5s	67.5s
GA2	*g size*=50, *#generations*=150	7500	61.5	58.5s	65.5s
GA3	*g size*=50, *#generations*=150	7500	56s	53.5s	58.5s
PS	*w*=0.25, *cp*=1.25, *cg*=3.5	7500	56s	53.5s	59s
TS(5)	*tau*= 10, *k*= changed(45 to 1)	7500	56s	52.5s	59s

metaheuristic algorithms' results before using benchmark iterative approach and the results of traditional methods were like Webster and HCM, and SYNCHRO simulator.

At the end, we can say that using metaheuristic optimization algorithms with the benchmark iterative approach, to optimize traffic light signals timing problem, were more suitable than using metaheuristic optimization algorithms without benchmark iterative approach and the traditional methods. Their results were less and were close to the optimal solution.

To verify the main conclusion, obtained from using the benchmark iterative approach in this research (see chapter 5), the most suitable algorithms for solving the optimization traffic light signals timing problem were GA Type 3, PS algorithm at *w*=0.25, *cg*=3.5 and *cp*=1.25, and TS Type 5 algorithm at *tau*=10, with 7,500 execution times. They had the minimum average (*ATT.*) for each vehicle. To make sure this conclusion has a good confidence, statistical tests (T test and confidence intervals) were used.

T test was used between two different samples to check whether the two means of the two samples were equal or not and which one was the best [36]. The steps of this test are as follows:

1- Start from the hypothesis

Null hypothesis: H_0 : $M1 = M2$, two means are equal.

Alternative hypothesis: H_1 $M1 \neq M2$, two means are not equal.

2- Determine error ratio Alfa = 0.05 and the significant value = 95%.

3- Compute the *(t)*value between the two samples through this equation [36]:

$$t = \frac{M_1 - M_2}{\sqrt[2]{\left(\frac{(n_1-1)S_1^2 + (n_2-1)S_2^2}{(n_1+n_2-2)}\right)\left(\frac{1}{n1}+\frac{1}{n2}\right)}} \quad \text{........(6.1).}$$

4- Compute the standard deviation (*S)* for each sample through this equation [36]:

$$S = \sqrt[2]{\frac{1}{n-1}\sum_{i=1}^{n}(x_i^2 - M)^2} \quad \text{............(6.2) .}$$

5- Get $t_{0,\frac{\alpha}{2}.df}$ from T table [36] p. 773 to see if H_0 is rejected or not [36].

6- Accept Null hypothesis if : $-\ t_{0,\frac{\alpha}{2}.df} \leq t \leq t_{0,\frac{\alpha}{2}.df}$

7- Reject Null hypothesis if: $-\ t_{0,\frac{\alpha}{2}.df} > t$ or $t > t_{0,\frac{\alpha}{2}.df}$

In this part, all experiments' results had a normal distribution as an assumption, because the average of the results for each algorithm almost centered in the middle of results. The algorithm GA Type 3 results with 7,500 execution times were found to be the best algorithm from all algorithms. T test was done between GA Type 3 results and all other algorithms' results as shown below:

1- Null hypothesis was: H_0 : $M1 = M2$

where *M1 is the means of GA3 results with 7,500* execution times were assumed to be the best, and *M2* is the means of each other algorithm results.

2- Alternative hypothesis: H_1 $M1 \neq M2$, two means are not equal.

3- Standard deviations were computed for each algorithm results as shown in Table 6.3 (see Appendix 2).

4- T values were computed between GA 3 results and each other algorithm results. Comparisons for each T value with $t_{\frac{0.05}{2},df}$, $t_{\frac{0.10}{2},df}$, $t_{\frac{0.20}{2},df}$ and $t_{\frac{0.50}{2},df}$ were done.

5- T test results show that the null hypothesis H_0 was rejected between GA3 results with 7,500 execution times and 58 states of other algorithms' results at significant level (95%) as shown in Table 6.3 (see Appendix 2). The best one was GA Type 3 algorithm because it had the minimum average (*ATT*) from all 62 states of other algorithms' averages (*ATT*). However, the null hypothesis was accepted just for GA Type 3 and PS algorithm at *w*=0.25, *cg*=3.5 and *cp*=1.25, and TS Type 5 algorithm at *tau*=10, with 7,500 execution times at significant level 95% as shown in Table 6.3 (see Appendix 2). That means the three algorithms were equal in the average (*ATT*) and were the best algorithms for optimizing the traffic light signals timing problem because they had the minimum average (*ATT*).

To test the average of each algorithm results of ATT. for each vehicle, represented with some reliability level, a confidence interval with significant level 95% was computed for each algorithm results (ATT) for each vehicle results, using this equation[36]:

$$M - t_{0.05/2,\ df} * \frac{s}{\sqrt[2]{n}} \leq \ \mu \ \leq M + t_{\alpha 0.05/2,\ df} * \frac{s}{\sqrt[2]{n}} \quad \ldots..(6.3).$$

Where μ is any new experiment result (ATT) located inside the confidence interval.

The confidence interval for each algorithm state is shown in Table 6.3 (see Appendix 2). The best three algorithms were GA Type 3 and PS algorithm, at *w*=0.25, *cp*=1.25 and *cg*= 3.5 and TS type 5 algorithm at *tau*=10, with 7,500 execution times; they had almost the minimum and the same confidence intervals as follow:

a- The confidence interval, with significant level 95% of the GA Type 3 algorithm with 7,500 execution times, was between 55.4s, and 56.5s.

b- The confidence interval, with significant level 95% of the PS algorithm at *w*=0.25, *cp*=1.25 and *cg*=3.5, with 7,500 execution times, was between 55.3s, and 56.5s.

c- The confidence interval, with significant level 95% of the TS Type 5 algorithm at *tau*=10, with 7,500 execution times, was between 54.9s, and 57.6s.

To conclude, after results of T test and the confidence intervals, it was found that the most suitable algorithms for optimizing the traffic light signals timing problem were GA Type 3, PS algorithm, at *w*=0.25, *cg*=3.5 and *cp*=1.25, and TS Type 5 algorithm, at *tau*=10, with 7,500 execution times. The benchmark iterative approach helped us to determine the best metaheuristic optimization algorithm' parameters' values to arrive at the best performance for the metaheuristic algorithm in a short time, thus optimizing the traffic light signals timing problem.

Chapter seven

Conclusion and Future Work

Metaheuristic optimization techniques are more suitable for solving the traffic light signals timing optimization problem than traditional and mathematical methods. Using the benchmark iterative approach in this research helped us in determining the best parameters' values as illustrated in the first chapter in short time. The metaheuristic optimization algorithm was more efficient at these values to get optimal or near optimal solution. Finding the optimal solution or near optimal solution in any optimization problem, selection of a suitable optimization algorithm for solving this problem are very important. Because some optimization algorithms especially in metaheuristic algorithms may be more efficient to get the optimal or near optimal solution than others with the same problem.

In this research, when many metaheuristic optimization techniques were used, to optimize the traffic light signals timing problem, some algorithms were efficient to get optimal or near optimal solution, but some other algorithms were not efficient and the results were far from the optimal solution, and sometimes their results were worse than the random algorithm.

The results of this research seem to be promising as illustrated in the previous two chapters. GA Type 3, PS, at *w*=0.25, *cg*=3.5 and *cp*=1.25, and TS Type 5 at *tau*=10 algorithms with 7,500 execution times were more efficient to get less possible outcomes out of the average travel time (*ATT*) for each vehicle than other algorithms states. By using these three algorithms the total waiting time at the traffic light signals, consuming petrol, and air pollution from the vehicles were reduced to a lower level indirectly.

New types of Tabu Search algorithm were improved in this research, such as TS types 3, 4 and 5 algorithms, were more suitable in solving both benchmark function and traffic light signals problem than the basic algorithms' types TS 1 and TS 2. They got the global minima value of benchmark function $y= 0$, and the minimum results of the average travel time (*ATT*) for each vehicle were less than the minimum algorithms' results of TS 1 and TS 2. Because the new types depended on the search in the farther neighborhoods, then in the nearest neighborhoods, but these new methods need to be proved mathematically in future research.

Some metaheuristic optimization algorithms are suitable for solving some optimization problems but may be very bad in solving other problems, and the topology of the problems' function may be the central point in determining which algorithm is the best, and this needs further research.

All research questions were answered; the research problem was solved. In the light of these results, the researcher recommends using benchmark iterative approach by the optimization researchers for optimal use of metaheuristic optimization techniques to solve the complex practical problems and find out the optimal or near optimal solution. He also recommends that roads engineers use the metaheuristic optimization algorithms with this approach to get the optimal or near optimal timing for their road network traffic light signals timing optimization problem.

At the end of this research, municipalities and other institutions, in charge of the road networks and the traffic light systems, can benefit from the results of this research by using the most suitable metaheuristic algorithms whose results were the best in this research. They can compute the average travel time for each vehicle for a statistical sample of vehicles or the flow of vehicles by using counters at the traffic light positions in the real world, with their optimal timing, before using the metaheuristic optimization algorithms, and the optimal timing for the traffic light signals after using the best metaheuristic optimization algorithms. Finally, they can then compare the results to see how much the best metaheuristic algorithm have improved the results of the average travel time for each vehicle and the flow of vehicles.

In the future research, the researcher suggests trying other types of metaheuristic optimization algorithms to solve traffic light signals problem in addition to other road networks with a high number of traffic light signals and other simulators for presenting the network. This may help in finding the most suitable and stable algorithm to get the optimal solution of the main research problem in the short time. Algorithms used in this research were very time consuming to reduce waiting time at traffic light signals; petrol consuming and air pollution from the vehicles would drop to lower levels.

References:

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, No. 6, pp. (125-132), 2013.
2. Yun I., Park B. (2006): Application of Stochastic Optimization Method for an Urban Corridor, Proceedings of the 2006 Winter Simulation Conference, 1-4244-0501-7/06/$20.00 ©2006 IEEE
3. Farooqi A., Munir A., Baig R.: THE: Traffic Light Simulator and Optimization Using Genetic Algorithm, International Conference on Computer Engineering and Applications IPCSIT, Vol.2, 2009.
4. Teklu F., Sumalee A., Watling D. : A Genetic Algorithm Approach for Optimizing Traffic Control Signals Considering Routing, Computer-Aided Civil and Infrastructure Engineering, vol. 22, pp.(31–43), 2007.
5. Singh L., Tripathi S., Arora H. : Time Optimization for Traffic Signal Control Using Genetic Algorithm, International Journal of Recent Trends in Engineering, Vol 2, No. 2, November 2009.
6. Renfrew D.: Traffic Signal control with Ant Colony Optimization. A Thesis presented to the Faculty of California Polytechnic State University, San Luis Obispo, (2009).
7. Singh D., Singh R.: Stochastic Optimization Method for Signalized Traffic Signal Systems, International Journal of Knowledge-Based and Intelligent Engineering Systems,Vol.13, pp.(71-71), (2009).
8. Sklenar J., Beranek J., Popela P.: Simulation-Based Heuristic Optimization of a Traffic System, Proceedings 23rd European Conference on Modeling and Simulation, 2009.
9. Singiresu R.: Engineering Optimization Theory And Practice, Third Edition, A Wiley-Interscience Publication, pp.(1-750), 1996.
10. Gentile G., Tiddi D.: Synchronization of traffic signals through a heuristic-modified genetic algorithm with GLTM. In Proceedings of the XIII Meeting of the Euro Working Group on Transportation, Padova University Press, Padua, Italy, pp(1-9), 2009.
11. Medina J., Moreno M., Cabrera M., Royo E.: Traffic Signals in Traffic Circles: Simulation and Optimization Based Efficiency Study. R. Moreno-D´ıaz et al. (Eds.): EUROCAST 2009, pp.(453–460), Springer-Verlag Berlin Heidelberg , 2009.
12. Abdalhaq B., Abu Baker M.: Using Meta Heuristic Algorithms to Improve Traffic Simulation, Journal of Algorithms and Optimization, Vol. 21ss. 4, PP. 110-128, (Oct. 2014).
13. Valle Y., Venayagamoorthy G., Mohagheghi S., Hernandez J. and Harley R.: Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems, IEEE Transactions on Evolutionary Computation, Vol.12, No.2, Pp.(171-195), 2008.
14. Garber J. N., Hoel A. L.: Traffic and Highway Engineering, Fourth Edition. Library of Congress,chap. 8, pp.(347-360), 2010.
15. Trafficware Synchro Website: http://www.Trafficware.Com/Products/Adaptive-Traffic-Control.

16. Abushehab R., Abdalhaq B., Sartawi B.: Genetic vs. Particle Swarm Optimization Techniques for Traffic Light Signals Timing, 2014 6th International Conference on csit, Jordan, Published by the ieee Computer Society, 2014.
17. H. Akbaripour and E. Masehian: Efficient and Robust Parameter Tuning for Heuristic Algorithms, International Journal of Industrial Engineering & Production Research, Volume 24, Number 2, pp. (143-150), 2013.
18. S.K. Smit and A.E. Eiben (2010): Parameter Tuning of Evolutionary Algorithms: Generalist vs. Specialist. Conference: Applications of Evolutionary Computation, Proceedings, Part I, Istanbul, Turkey, 2010.
19. XU J., CHIU S. and GLOVER F.: Fine-tuning a Tabu Search Algorithm with Statistical Tests, Published by Elsevier Science Ltd All rights reserved, Vol. 5, No. 3, pp. (233-244), 1998.
20. Balci H. and Valenzuela J.: Scheduling Electric Power Generators Using Particle Swarm Optimization Combined With The Lagrangian Relaxation Method, Int. J. Appl. Math. Comput. Sci., amcs, Vol. 14, No. 3, pp.(411–421), 2004.
21. Pedersen M. (2010): Good Parameters for Particle Swarm Optimization, Technical Report no. HL1001.
22. Mitchell M.: Genetic Algorithms: An Overview1, Complexity, Vol.(1) , Pp.(31-39), 1995.
23. L.B. Booker, D.E. Goldberg, and J.H. Holland: Classifier Systems and Genetic Algorithms, Published in journal Artificial Intelligence, Vol. 40, pp. (235-282), 1989.
24. Kennedy J. and Eberhart R.: Particle Swarm Optimization. Book Title Computational Intelligence PC Tools, published in 1996 by Academic Press Professional (APP), pp.(1942-1948), 1995.
25. Garc´Ia-Nieto J., Olivera A. and Alba E.: Optimal Cycle Program of Traffic Lights with Particle Swarm Optimization, 2013.
26. Glover F.: Tabu Search - Part I. Orsa Journal on Computing, Vol. 1, No. 3, pp.(190-206), 1989.
27. Glover F.: Tabu Search - Part I I. Orsa Journal on Computing, Vol. 2, No. 1, pp.(4-32), 1990.
28. Kelton D., Sadowski S., Sadowski D.: Simulation with Arena, Second Edition p.9.
29. Maria A.: Introduction to Modeling and Simulation. Proceedings of the Winter Simulation Conference, pp.(7-13), 1997.
30. Papageorgiou G., Damianou P., Pitsillides A., Aphamis T., Charalambous D., Ioannou P.: Modelling and Simulation of Transportation Systems: A Scenario Planning Approach. Automatika 50, vol.1, No.2, pp.(39-50), 2009.
31. Soh A., Khalid M., Yusof R., Marhaban M.: Modelling of a Multilane Traffic Intersection Based on Networks of Queues and Jackson's Theorem.
32. Sumo – Simulation Of Urban Mobility Website:(http://sumo.sourceforge.net).
33. Daniel K., Georg H. And Peter W. (2002): Sumo (Simulation Of Urban Mobility) An Open-Source Traffic Simulation.(http://sumo.sourceforge.net).
34. Python Programming Language Website: Http://Www.Parallelpython.Com/Content/View/15/30/. Parallel Python Community Parallel Python, 2015 Vitalii Vanovschi. All rights reserved.
35. Johann D., Alain P., Professor T.: Meta-Heuristics for Hard Optimization, Originally published in French by Eyrolles, Paris, Springer, pp.(83-91), 2003.
36. Douglas M., George R.: Applied Statistics and Probability for Engineers, Third Edition. Library of Congress Cataloging-in-Publication Data, Printed in the United States of America, pp.(66-396), 2003.

Appendix 1: Summary of all experiments results

Table 5.1: Results of Experiment 1, traffic light signals timing problem with SUMO simulator, 30 chromosomes and 50 generations.

algorithms'	*MP*	repeating experiments	*AVG. (ATT.)* for each vehicle	*Mini(ATT.)* for each vehicle	*Max. (ATT.)* for each vehicle
Random		20 times	67.8s	63s	72.6s
GA Type 1	10%	10 times	70.2s	68.7s	72.5s
	20%		70.5s	64.7s	73s
	30%		69.1s	64.9s	72.6s
	40%		67.9s	64.3s	71s
	50%		67.4s	65.7s	69.2s
	60%		67.9s	66s	70.3s
	70%		66.9s	63.9s	69.8s
	80%		67.8s	63.1s	72s
	90%		68.4s	64.8s	70.7s
GA Type 2	10%	10 times	63.1s	58.1s	66.5s
	20%		63.9s	60.3s	69s
	30%		62.6s	60.3s	64.6s
	40%		63.4s	60.8s	66.8s
	50%		64.5s	59.2s	67.7s
	60%		63.2s	56.8s	68.3s
	70%		62.5s	60.7s	64.9s
	80%		63.9s	62s	65.5s
	90%		64s	60.5s	67.7s
GA Type 3		20 times	60.8s	56.5s	67.3s

Table 5.2: Results of Experiment 2, benchmark function with 30 chromosomes and 50 generations.

algorithms'	*MP*	repeating experiments	*AVG.*(experiments results)	*Min.*(experiments results)	*Max.* (experiments results)
Random algorithm	0%	110 times	168	118.5	199.4
GA Type 1	10%	10 times	40.8	32.8	52.8
	20%		37.2	25	50
	30%		39.7	30	46.8
	40%		42.7	25	49.8
	50%		46	35.6	58.2
	60%		45.5	35.6	59.5
	70%		47.5	33.5	57
	80%		48	43	59
	90%		52.2	43.4	62.8
GA Type 2	10%	10 times	69.9	45.4	94.2
	20%		68	49	95.1
	30%		76.8	45.6	104.3
	40%		71.3	53.4	89.1
	50%		74.3	49.8	94.6
	60%		69.4	42.8	87.5
	70%		68.7	57.2	97.7
	80%		64.9	45.8	83.8
	90%		64.4	56	85
GA Type 3	0	20 times	16.1	7	27.8

Table 5.3: Results of Experiment 3, benchmark function with 50 chromosomes and 150 generations.

algorithms'	***MP***	**repeating experiments**	***AVG.*(experiments results)**	***Min.*(experiments results)**	***Max.* (experiments results)**
Random	0%	110 times	109.7	81.4	128.4
GA Type 1	10%	10 times	35.3	27.6	45.8
	20%		33.5	28.8	40.2
	30%		34.8	30.2	40.4
	40%		33.7	27.2	39.4
	50%		35.4	27.4	40.7
	60%		38.6	31.2	44.2
	70%		37.4	28.4	42.6
	80%		40.7	36	51.3
	90%		41.2	33.6	49.2
GA Type 2	10%	10 times	41.2	25.6	37.8
	20%		39.6	31	52.8
	30%		36	21.6	47.6
	40%		37.6	31.4	54
	50%		41.7	31	57.1
	60%		37.7	29.2	53.2
	70%		41.1	33.4	56.2
	80%		30.8	30.8	61.8
	90%		39.3	20.6	46.6
GA Type 3	0%	20	1.3	0	5
PS	0%	20	53.3	20.8	82.8

Table 5.4: Results of Experiment 4, traffic light signals timing problem with SUMO simulator, 50 chromosomes and 150 generations.

algorithms'	***MP***	**repeating experiments**	***AVG. (ATT.)* for each vehicle**	***Mini(ATT.)* for each vehicle**	***Max. (ATT.)* for each vehicle**
Random		10 times	64.4s	62.5s	67.6s
GA Type 1	10%		62.2s	58.4s	67.5s
	20%		59.2s	57.3s	60.5s
	30%		58.8s	56.5s	62.5s
	40%		58.2s	57.4s	59s
	50%		59.1s	56.3s	60.7s
	60%		61.1s	59.3s	63.9s
	70%		61.6s	59.6s	65.2s
	80%		61.8s	60.4s	63.7s
	90%		62.5s	60.4s	65.1s
GA Type 2	10%		60.4s	59s	61.8s
	20%		61.2s	58.5s	64.9s
	30%		60.8s	59.3s	63.6s
	40%		60.7s	58.6s	62.4s
	50%		61s	58.6s	62.9s
	60%		61.8s	59.7s	63.8s
	70%		62.6s	59.6s	65.4s
	80%		62s	58.7s	64.7s
	90%		61.6s	59.6s	63.5s
GA Type 3		20 times	55.9s	53.4s	58.3s
PS			58.9s	54.9s	71.2s

Table 5.5: Results of Experiment 5, PS algorithm in solving benchmark function with 50 chromosomes and 150 generations.

Probability #	*w*	*cg*	*cp*	# of Results res=<10	Probability of results <= 10	*Max.* results	*Min.* results
1	0	0.75	2.25	8	40%	18	6
2	0	0.75	2.5	11	55%	20	5
3	0	0.75	2.75	11	55%	26	6
4	0	0.75	3	8	40%	34	4
5	0.25	0.75	2.75	6	30%	20	4

Table 5.6: Results of Experiment 6, PS algorithm in solving traffic light signals timing problem with 50 chromosomes and 150 generations.

experiments	***w=***	***cp=***	***cg=***	**# of experiment repetition**	***AVG.(ATT)***	***Min.(ATT)***	***Max.(ATT)***
1	0	2.5	0.75	6	64.3s	61.8s	66.6s
2	0	2.75	0.75	6	62.3s	59.6s	66.4s
3	0.25	1.25	3.5	20	55.9s	53.4s	58.9s
4	0	0.75	3.5	20	58s	53.4s	69.5s
5	0	0.25	4.25	10	57.5s	53.2s	69.3s
6	0	0.25	4.75	10	60.2s	55.9s	66.7s

Table 5.7: Results of Experiment 7, TS Type1 algorithm in solving benchmark function with *tau*= {1, 2, 3, …, 12}, and *k*={0.1, 0.2, 0.3, …, 2.5}.

tau=	**1**			**2**			**3**			**4**			**5**			**6**			**7**			**8**			**9**			**10**			**11**			**12**		
k=	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***	***avg.***	***min.***	***max.***
0.1	64	28	94	76	44	102	74	51	98	69	43	93	70	41	105	72	43	94	64	34	82	68	38	101	67	44	99	66	38	106	72	53	93	74	52	103
0.2	90	69	118	78	56	110	79	49	109	83	58	106	81	59	119	66	29	106	71	30	103	75	52	97	79	48	116	77	45	116	85	68	106	85	61	102
0.3	89	48	127	89	71	115	92	55	122	87	71	100	92	68	118	92	77	110	93	60	117	85	57	111	78	54	95	86	71	114	80	67	88	77	47	107
0.4	65	40	93	66	43	100	78	64	89	73	54	102	89	73	125	94	72	116	96	54	131	83	67	125	99	85	121	93	63	109	91	34	122	94	63	111
0.5	87	33	139	72	44	99	68	39	101	65	33	90	50	39	94	51	28	84	47	17	59	50	31	80	50	28	66	48	40	56	42	24	52	41	21	67
0.6	65	41	97	64	41	112	69	49	84	74	53	115	86	44	125	84	58	119	82	62	102	79	55	107	86	55	123	83	65	93	83	63	103	87	63	108
0.7	80	49	111	75	57	122	82	59	111	72	53	112	76	40	138	78	52	102	82	40	111	73	55	115	56	44	63	59	45	76	66	45	87	73	53	108
0.8	83	56	121	71	43	96	68	35	99	64	40	91	60	30	79	62	42	85	60	28	112	76	45	104	62	40	80	59	41	89	66	39	88	74	38	116
0.9	67	52	102	61	27	99	65	39	91	69	33	94	68	36	109	73	42	94	76	37	102	61	35	106	66	35	96	73	53	103	68	39	103	57	25	82
1.0	82	42	111	76	61	100	81	70	99	82	58	102	89	66	117	86	44	123	89	45	121	88	69	111	89	59	118	80	46	107	83	64	100	81	52	112
1.1	78	54	95	68	52	83	73	51	96	72	48	97	62	38	99	73	54	92	77	36	105	75	50	114	73	44	95	79	46	124	78	62	94	72	37	94
1.2	61	35	89	56	43	81	53	34	80	53	41	74	54	27	86	54	31	78	51	37	62	61	44	94	66	49	96	59	35	87	67	53	85	62	32	85
1.3	55	33	93	54	34	74	52	38	77	57	40	77	51	36	76	57	40	79	50	38	67	53	33	69	55	34	78	61	42	82	60	48	73	59	45	92
1.4	63	38	94	54	34	73	52	41	70	53	39	83	51	38	82	58	33	70	52	26	66	55	27	81	51	41	60	53	42	72	56	46	67	59	41	75
1.5	72	32	100	56	29	95	54	29	76	46	27	61	49	29	61	50	30	66	48	37	63	53	43	65	52	40	79	55	45	67	56	47	74	55	38	79
1.6	52	43	61	51	30	75	52	42	86	52	31	69	56	37	74	54	41	65	56	39	68	57	42	70	55	34	71	49	38	62	58	36	70	50	23	79
1.7	51	35	70	50	37	70	56	39	71	49	37	61	57	43	67	64	48	107	59	40	77	62	45	86	65	50	91	60	45	99	63	42	79	58	35	88
1.8	69	55	85	66	36	88	70	36	89	69	43	94	73	47	85	83	58	118	66	45	87	80	62	106	74	43	112	71	38	104	62	47	84	76	55	96
1.9	96	66	123	91	75	111	90	57	124	86	40	108	88	68	149	84	58	108	88	49	134	95	74	124	86	71	120	80	52	134	94	72	129	89	49	119
2.0	91	52	109	99	75	124	91	61	114	83	52	111	92	61	122	95	76	128	87	62	108	97	63	134	83	47	125	89	64	104	85	71	106	81	58	107
2.1	94	58	115	105	74	138	76	42	106	95	25	138	103	79	136	83	66	109	88	69	106	88	65	108	92	64	115	88	45	145	83	65	98	87	50	119
2.2	76	52	113	80	50	106	81	53	130	89	71	108	86	55	114	79	53	130	90	63	109	78	55	126	76	61	97	83	54	111	82	58	105	79	53	103
2.3	65	42	89	65	33	90	74	62	83	75	49	92	68	48	85	74	53	107	74	56	90	71	56	96	72	54	97	76	61	90	72	59	85	71	48	93
2.4	77	56	96	68	42	88	65	52	88	69	51	92	65	47	88	67	34	91	61	45	90	64	51	81	69	55	79	66	43	89	78	61	103	72	60	90
2.5	47	24	109	46	24	87	60	34	101	53	38	67	65	49	96	60	41	75	65	51	77	66	48	75	71	51	86	68	43	93	65	40	83	69	42	93

Table 5.8: Results of Experiment 7, TS Type 2 algorithm in solving benchmark function with *tau*= {1, 2, 3, …, 12}, and *k*={0.1, 0.2, 0.3, …, 2.5}.

Tau=	**1**			**2**			**3**			**4**			**5**			**6**			**7**			**8**			**9**			**10**			**11**			**12**		
time	**10**			**10**			**10**			**10**			**10**			**10**			**10**			**10**			**10**			**10**			**10**			**10**		
K=	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*	*avg.*	*min.*	*max.*
0.1	75	45	116	72	48	103	77	55	105	76	43	121	69	40	97	70	46	98	68	35	95	72	56	94	83	62	121	70	42	117	70	47	98	71	27	113
0.2	85	66	105	78	51	104	73	57	97	83	52	108	78	62	101	75	34	100	83	58	117	74	49	110	85	43	135	74	38	111	82	42	102	79	61	101
0.3	86	50	117	93	65	119	87	49	120	90	65	120	90	64	111	76	43	113	84	46	113	79	46	96	81	63	108	78	59	112	89	56	113	80	53	111
0.4	71	27	106	59	37	80	72	47	87	75	61	105	82	60	117	85	60	110	78	32	113	84	62	118	88	54	122	72	57	93	82	70	94	80	47	121
0.5	77	33	108	72	32	122	64	22	83	50	30	91	54	41	82	54	31	83	63	40	93	50	31	65	47	22	72	51	37	71	41	28	62	38	28	53
0.6	48	26	90	66	40	100	57	32	86	70	27	104	73	47	126	68	35	99	59	36	95	58	33	83	74	53	99	75	56	112	64	37	107	64	47	88
0.7	83	60	118	75	57	99	77	43	112	82	57	125	78	50	99	91	67	127	73	42	111	58	32	107	62	38	93	62	45	95	71	50	90	73	49	92
0.8	67	33	85	63	39	75	69	32	97	69	46	92	73	52	140	72	34	120	72	42	118	53	32	68	57	44	76	59	46	70	57	28	76	62	45	82
0.9	68	33	91	66	46	86	67	49	90	65	54	78	66	36	83	65	33	98	64	48	81	68	42	103	66	40	91	74	34	112	66	25	95	67	44	84
1	82	59	116	90	55	137	88	74	104	84	56	110	92	60	130	84	41	103	89	61	117	82	62	98	88	55	107	85	62	103	90	75	114	75	46	98
1.1	67	35	100	68	58	79	72	46	90	66	30	96	69	32	87	80	52	117	68	41	93	84	65	106	67	44	94	70	50	92	78	57	105	74	46	97
1.2	66	29	122	56	40	69	52	29	79	47	29	75	47	28	66	54	35	74	62	42	71	60	44	74	60	28	80	60	32	75	64	44	103	64	48	76
1.3	55	36	108	51	33	68	55	26	115	57	36	82	53	31	63	48	32	62	56	27	69	65	45	95	58	38	81	62	31	85	66	44	90	62	35	87
1.4	54	40	67	52	36	80	45	35	62	55	41	69	56	38	73	58	33	78	62	36	82	56	46	71	52	26	82	52	27	73	53	31	65	56	33	67
1.5	58	39	95	59	35	96	63	24	90	49	33	80	48	29	67	39	29	53	49	30	80	38	21	51	45	20	58	43	29	66	44	25	62	48	32	60
1.6	49	30	66	58	37	92	49	31	60	55	36	74	51	30	84	51	33	59	48	27	68	44	33	65	45	37	54	53	39	74	56	31	66	57	38	74
1.7	53	31	73	57	46	72	44	34	67	55	37	71	51	30	75	64	48	84	55	31	74	54	30	77	68	36	99	63	42	87	59	35	87	54	35	66
1.8	72	54	85	73	54	92	71	50	96	74	63	106	67	44	101	67	44	93	69	48	109	77	59	90	80	54	102	74	52	106	76	55	102	76	49	107
1.9	79	54	96	83	32	112	92	62	112	91	55	113	96	74	141	84	61	102	82	32	108	86	51	134	83	64	112	76	42	108	86	69	113	92	60	117
2	86	57	116	97	59	131	97	75	118	94	70	140	101	84	134	80	63	103	93	65	107	94	46	121	93	54	138	100	84	117	92	60	116	102	69	136
2.1	87	38	125	87	44	110	89	48	117	87	57	118	92	67	138	93	57	122	88	56	124	83	53	119	97	79	128	91	63	119	94	49	130	70	40	98
2.2	88	61	109	70	50	90	78	59	98	74	53	122	73	57	99	81	46	121	82	64	109	72	49	114	89	70	116	75	57	96	79	41	99	81	61	101
2.3	70	57	90	70	57	80	68	47	91	68	48	87	71	57	95	65	54	73	74	56	89	75	64	90	69	43	87	74	51	93	70	56	82	70	51	89
2.4	68	47	98	71	62	83	73	61	98	69	56	89	66	35	98	64	47	92	67	52	83	68	58	82	68	50	85	69	53	93	65	50	83	69	58	83
2.5	41	26	86	59	33	96	52	35	102	56	39	79	56	38	66	66	39	80	62	38	81	64	45	82	62	45	82	58	36	80	60	25	84	48	31	60

Table 5.9: Results of Experiment 7, TS Types 3, 4 and 5 algorithms in solving benchmark function with 7538 execution times.

tau		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
time repeating		**50**	**50**	**50**	**50**	**50**	**50**	**50**	**50**	**50**	**50**	**50**	**50**
Tabu type 3	**Result =0**	37	42	40	24	13	36	33	1	1	0	0	0
	Mini.	0	0	0	0	0	0	0	0	0	2	7	5
	Max.	2.8	2	2	2.8	6	3	5	10	9	19.8	19.6	30.6
	AVG.	0.3	0.2	0.28	0.85	2.7	0.6	0.87	6.1	5.3	11.5	13.8	13.9
Tabu type 4	**Result =0**	7	49	27	17	8	40	32	0	0	0	0	0
	Mini.	0	0	0	0	0	0	0	4	2	4	6	6
	Max.	3	2	2	3	6	6.8	6.8	9.8	11	21.6	23.8	34.6
	AVG.	1.34	0	0.52	0.95	3.2	0.7	1.1	6.3	8.2	11.1	14.8	15
Tabu type 5	**Result =0**	50	50	50	50	50	50	50	50	50	50	50	50
	Mini.	0	0	0	0	0	0	0	0	0	0	0	0
	Max.	0	0	0	0	0	0	0	0	0	0	0	0
	AVG.	0	0	0	0	0	0	0	0	0	0	0	0

Table 5.10: Results of Experiment 8, TS Types 1,2,3,4 and 5 algorithms in solving traffic light signals timing optimization problems with SUMO simulator with 7538 execution times.

tau =	2			4			6			8			10		
time repeating	10			10			10			10			10		
results	*Avg.*	*Min.*	*Max.*	*Avg.*	*Min.*	*Max.*	*Avg.*	*Min.*	*Max.*	*Avg.*	*Min.*	*Max.*	*Avg.*	*Min.*	*Max.*
Tabu type1 at *k*=7							71.1s	64.3s	77.9s						
Tabu type2 At *k*=7							72.4s	64.5s	82.4s						
Tabu type3	64.8s	58s	69s	62.7s	57.8s	68.5s	63s	56.9s	70s	61.8s	57.1s	73s	63.2s	56.9s	68.6s
Tabu type4	59.9s	56s	66.3s	57.3s	53.7s	62.6s	58.8s	56.5s	61.2s	63.1s	58.9s	70.2s	61.2s	57.1s	66.5s
Tabu type5	58.4s	54.4s	65.3s	58.3s	54.3s	68.8s	57.2s	53.4s	64.5s	57.2s	55.2s	59.2s	56.2s	52.5s	59.1s

Appendix 2: Validity and reliability of all experiments results

Table 6.3: Results reliability by t-test between GA3 with 7500 execution times and all algorithm results at significant 95%, and by confidence interval with significant 95% for each algorithm.

experiments	algorithms	conditions	*N*=	Means=	*STD*=	confidence interval with significant level= 95%		H_0: $M1 = Mn$ t test between GA 3 with 7500 iteration and each algorithm results at significant level=
						lower =	upper=	95%
experiment(1) traffic light signals timing problem with SUMO simulator with 1500 execution times	rand.		20	67.8	10.8	66.5	69.1	rejected
	GA 1	***MP*=10%**	10	70.2	1.4	69.3	71.0	rejected
		***MP*=20%**	10	70.5	2.5	68.9	72.0	rejected
		***MP*=30%**	10	69.1	2.4	67.7	70.6	rejected
		***MP*=40%**	10	67.9	2.1	66.7	69.2	rejected
		***MP*=50%**	10	67.4	1.4	66.5	68.2	rejected
		***MP*=60%**	10	67.9	1.4	67.0	68.8	rejected
		***MP*=70%**	10	66.9	1.8	65.7	68.0	rejected
		***MP*=80%**	10	67.8	2.8	66.1	69.5	rejected
		***MP*=90%**	10	68.4	2.0	67.2	69.6	rejected
	GA 2	***MP*=10%**	10	63.1	3.4	61.0	65.3	rejected
		***MP*=20%**	10	63.9	2.6	62.3	65.5	rejected
		***MP*=30%**	10	62.6	1.4	61.7	63.5	rejected
		***MP*=40%**	10	63.4	1.9	62.3	64.6	rejected
		***MP*=50%**	10	64.5	2.5	63.0	66.1	rejected
		MP=60%	10	63.2	3.3	61.1	65.2	rejected
		***MP*=70%**	10	62.5	1.7	61.5	63.5	rejected
		***MP*=80%**	10	63.9	0.9	63.3	64.4	rejected
		***MP*=90%**	10	64.0	2.2	62.7	65.4	rejected
	GA 3		20	60.8	9.4	59.4	62.2	rejected
experiments(4, 6 and 8) traffic light signals timing problem with SUMO simulator with 7500 execution times	rand.		10	64.4	1.6	63.4	65.4	rejected
	GA 1	***MP*=10%**	10	62.2	2.9	60.4	64.0	rejected
		***MP*=20%**	10	59.2	1.0	58.6	59.8	rejected
		***MP*=30%**	10	58.8	1.7	57.8	59.9	rejected
		***MP*=40%**	10	58.2	0.8	57.7	58.7	rejected
		***MP*=50%**	10	59.1	1.2	58.3	59.9	rejected
		***MP*=60%**	10	61.1	1.4	60.2	62.0	rejected
		***MP*=70%**	10	61.6	1.9	60.4	62.8	rejected
		***MP*=80%**	10	61.8	1.0	61.2	62.4	rejected
		***MP*=90%**	10	62.5	1.5	61.5	63.4	rejected
	GA 2	***MP*=10%**	10	60.4	1.0	59.8	61.0	rejected
		***MP*=20%**	10	61.2	1.7	60.1	62.2	rejected
		***MP*=30%**	10	60.8	1.4	59.9	61.7	rejected
		***MP*=40%**	10	60.7	1.4	59.9	61.6	rejected
		***MP*=50%**	10	61.0	1.5	60.1	61.9	rejected
		***MP*=60%**	10	61.8	2.1	60.5	63.1	rejected
		***MP*=70%**	10	62.6	1.7	61.5	63.6	rejected
		***MP*=80%**	10	62.0	2.0	60.7	63.2	rejected
		***MP*=90%**	10	61.6	1.2	60.8	62.3	rejected

algorithm	conditions	*N*=	Means=	*STD*=	confidence interval with significant level=95%		H_0: *M1= Mn* t test between GA 3 50ch 150g and each algorithm results at significant
					upper=	lower=	95%
GA 3		20	55.9	7.9	55.4	56.5	
PS	***w*=0.25 *cp*=1 *cg*=2**	20	57.0	9.4	57.1	60.7	rejected
	***w*=0 *cp*=2.5 *cg*=0.75**	6	64.3	1.9	62.8	65.8	rejected
	***w*=0 *cp*=2.75 *cg*=0.75**	6	62.3	2.7	60.1	64.4	rejected
	***w*=0.25 *cp*=1.25 *cg*=3.5**	20	55.9	7.9	55.3	56.5	accepted
	***w*=0 *cp*=0.75 *cg*=3.5**	20	58.0	3.9	56.3	59.7	rejected
	***w*=0 *cp*=0.25 *cg*=4.25**	10	57.5	5.1	54.3	60.7	accepted
	***w*=0 *cp*=0.25 *cg*=4.75**	10	60.2	3.6	57.9	62.5	rejected
TS 1	***tau*=6 *k*=7**	10	71.1	4.8	68.1	74.0	rejected
TS 2	***tau*=6 *k*=7**	10	72.4	6.1	68.6	76.2	rejected
TS 3	***tau*=2 , *k* changed number**	10	64.8	3.3	62.7	66.8	rejected
	***tau*=4 , *k* changed number**	10	62.7	3.6	60.5	64.9	rejected
	***tau*=6 , *k* changed number**	10	63.0	4.7	60.1	65.9	rejected
	***tau*=8 , *k* changed number**	10	61.8	5.0	58.7	64.9	rejected
	***tau*=10 , *k* changed number**	10	63.2	3.7	60.9	65.5	rejected
TS 4	***tau*=2 , *k* changed number**	10	59.9	3.3	57.9	62.0	rejected
	***tau*=4 , *k* changed number**	10	57.3	2.7	55.6	59.0	accepted
	***tau*=6 , *k* changed number**	10	58.8	1.5	57.9	59.7	rejected
	***tau*=8 , *k* changed number**	10	63.1	4.1	60.6	65.6	rejected
	***tau*=10 , *k* changed number**	10	61.2	3.4	59.1	63.2	rejected
TS 5	***tau*=2 , *k* changed number**	10	58.4	3.6	56.2	60.6	rejected
	***tau*=4 , *k* changed number**	10	58.3	5.1	55.2	61.5	accepted
	***tau*=6 , *k* changed number**	10	57.2	3.3	55.2	59.2	accepted
	***tau*=8 , *k* changed number**	10	57.2	1.4	56.3	58.1	rejected
	***tau*=10 , *k* changed number**	10	56.2	2.1	54.9	57.6	accepted