Australian Journal of Basic and Applied Sciences, 3(4): 4333-4343, 2009
ISSN 1991-8178

Design and Optimization of a Traffic Control System by Integration of Computer Simulation and Genetic Algorithm

Ali Azadeh, PhD and Amin Sadeghnia, MS

Department of Industrial Engineering and Center of Excellence for Intelligent Based Experimental Mechanics, Collage of Engineering, University of Tehran, P.O. Box 11365-4563, Iran

Abstract: In this paper, simulation and Genetic Algorithm (GA) are integrated and used for optimization of a two direction traffic system with closed section for repair activities. The main application of the integrated approach is for design of intelligent traffic control system. The waiting times for green lights depend on the rate of traffic. First, computer simulation is used to model the traffic system. Then, simulation is integrated with GA to evaluate waiting times and time in system in the two directions. The average waiting time per vehicle and consequently the time in system are considerably lower than conventional simulation approach. This is shown through independent t-test. The proposed integrated approach may be used for design of intelligent traffic systems whose status changes during the day. This is the first study that introduces an integrated GA-Simulation approach for improvement of traffic systems with closed section for repair activity.

Key words: Computer Simulation; Genetic Algorithm (GA); Traffic Control; Optimization

INTRODUCTION

Computer simulation is one of the most advanced and powerful tools in system analysis. The simulation approach would enable the designers and analysts to foresee the behavior of such systems in normal and also emergency situations. Undoubtedly, simulation approach leads to a smoother and more efficient performance for such systems. In highly industrialized countries; computer simulation has become one of systems' performances. One of the major applications for simulation is optimizing existent systems. In fact an existent system simulated could examine easily different alternatives that exist for those and promote the application of system (Goldberg, 1989). Simulation could significantly help in modeling and control of traffic. For instance, we can simulate an intersection by using simulation and then determine proper time for green and red lights in order to minimize the congestion. There have been several studies in the field of traffic control and GA. A recent study obtained a better timer design for a pre-timed traffic signal of an intersection located in an urban area by using computer simulation. The objective was to reduce the average waiting time per approaching vehicle during peak hours by redesigning the current signal timer (Chou *et al*, 2001). Consequently, a better signal timer design is proposed. If the signal timer is appropriately set, the average waiting time per vehicle during peak hours is expected to have a reduction of about 20.74%. GA is used with traffic samples using real world alternative routes and a dynamic approach that tries to match the daily dynamic planning operations to take into account the stochastic demand. A first comparison between the static and the dynamic approach is performed. Two case studies investigated the applicability of genetic algorithms in complex optimization problems. The first case study concerns the use of GA to support planning in Air Traffic Management (ATM). Finding a route for conflicting aircraft is done by the genetic algorithm. The second case study concerns the use of GA for designing wing. The objective function combines (using different weights) the drag coefficient and penalties on thickness, lift coefficient, and separation of the flow at the upper and lower trailing edge of the airfoil (Oussedik, *et al*, 1999). Network operators must have control over the flow of traffic into, out of, and across their networks. Usually, changes of inter domain traffic affect intra domain traffic in other domains. Using the data of inters domain traffic and intra domain traffic measured in transit networks, the approach evaluates the cost of transit networks. Genetic algorithm has been presented to specify a link for prefix in AS neighbors with the objective of minimizing costs and configuration changes

Corresponding Author: Ali Azadeh, PhD., Department of Industrial Engineering and Center of Excellence for Intelligent Based Experimental Mechanics, Collage of Engineering, University of Tehran, P.O. Box 11365-4563, Iran
E-mail: aazadeh@ut.ac.ir

(Wang, *et al*, 2005). Another study introduced a versatile traffic micro simulation model for the rural roads which is capable of handling all common types of rural roads, including the effects of roundabouts and intersections on the traffic on the main road (Tapani, 2005). Cameron and Duncan developed road traffic simulation for the Edinburgh Parallel Computing Centre (Cameron, *et al*, 1996). Chen and Kun developed a dynamic traffic simulation system for accessing the road network reliability under large-scale events by dynamic traffic simulation model integration and road network reliability evaluation methodologies (Kun, *et al*, 2008). Yang and Jianguo presented a pedestrian model for traffic system micro-simulation in China (Jianguo, *et al*, 2006). A recent paper is concerned with the movement of vehicles around a road junction. It is desirable to set the time in a traffic light control system for the junction optimally for efficient use of the roads served by the junction. A simulation model is designed to study the problem. It was found that the average queue length in the system is directly proportional to the total phase length (Ayeni, 1990). Wen and Yang designed a framework for a dynamic and automatic traffic light control system and developed a simulation model to help design the system (Wen and Yang, 2006). Another study discussed a special-purpose simulation (SPS) tool for optimize traffic signal light timing. The simulation model is capable of optimizing signal light timing at a single junction as well as an actual road network with multiple junctions. It also provides signal light timing for certain time periods according to traffic demand (Kasun and Janaka, 2004). Sadoun analyzed and demonstrated how the timings of a traffic light system are calculated in order to determine the optimum and proper timings needed to optimize traffic flow (Sadoun, 2008).

The electro sensitive traffic light using fuzzy look up table method was proposed which will reduce the average vehicle waiting time and improve average vehicle speed (Gi Young,, *et al*, 2002). Jungblut-Hessel presented a method to perform fast simulation of large Markovian systems. This method was based on the use of three concepts: Markov chain uniformization, event-driven dynamics, and modularity (Jungblut, *et al*, 2001). De Schutter and De Moor considered an intersection of two two-way streets with controllable traffic lights on each corner. They constructed a model that described the evolution of the queue lengths (as continuous variables) in each lane as a function of time. They discussed how optimal and suboptimal traffic light switching schemes (with possibly variable cycle lengths) for this system can be determined (De Schutter, *et al*, 1998). Sanchez presented a new architecture for the optimization of traffic light cycles in a traffic network. The model is based on three basic design items: The use of Genetic Algorithms as an optimization technique, the use of Cellular Automata Simulators within the Evaluation Function, and the use of a Beowulf Cluster as parallel execution environment for this architecture (Sanchez and Rubio, 2004). Zhao *et al.* selected a single intersection of two phases as a model to put forward a new optimal time-planning scheme for traffic light based on the model of hybrid automata for single intersection. A method of optimization is proposed for hybrid systems, and the average queue length over all queues is used as an objective function to find an optimal switching scheme for traffic light. It is illustrated that traffic light control for single intersection is a typical hybrid system, and the optimal planning-time scheme can be obtained using the optimal hybrid systems control based on the two stages method (Zhoa and Cui, 2003). Huang and Chung focused on the use of Timed Colored Petri nets(TCPNs) to model an intelligent urban traffic light control system. The intelligent traffic light controller proposed a solution to regulate the problem of traffic congestion. For this purpose, in this paper also presented an intelligent control methodology which was constructed by the alternation of two different sub-models (Huang and Chung, 2009).

A genetic algorithm is one of a class of algorithms that searches a solution space for the optimal solution to a problem. The algorithm creates a "population" of possible solutions to the problem and lets them "evolve" over multiple generations to find better and better solutions. A "population" of possible solutions is formed, and new solutions are formed by "breeding" the best solutions from the population's members to form a new generation. Consequently, the best solution is returned by GA. This paper examines a problem related to traffic control systems. This problem is simulated by Visual SLAM and optimal waiting times for lights are calculated by GA method to minimize the time in system. In fact, the goal of simulation is to decrease time in system and increase system operation.

Genetic Algorithm:

Genetic algorithms are practical, robust, optimization and stochastic search algorithms based on the mechanics of natural genetics. Genetic algorithms were first developed by Holland in 1975. They use the Darwinian principle of survival-of-the fittest to eliminate unfit characteristics. The searching process is similar to the natural evaluation of living things. The fittest among a group of creatures can survive and form a new generation randomly and by gene exchange. Adaptation to a changing environment is essential for the survival of each species. Genetic algorithms use stochastic processes to produce an initial population of models which

is called parents. Then, a new population is produced through iteration. Each iteration represents a new generation. In every new generation, a new set of strings is created according to a number of specified performance indices. GA searchs for new generations with gradually improved behavior. This process is repeated a number of times until the best model evolve. For an optimization problem, process model is represented by simple encoding. Here, the parameters are coded mostly into binary digits or bit strings. Each coded solution, which consists of a set of unknowns, $x_i, i = 1,...,M$ is represented by a vector termed a chromosome. Fitness function provides the mechanism for evaluating each string. Having defined the initial population and the fitness function, the parameters are evaluated and assigned with a fitness value which shows how good is the selected solutions compared with others in the population. Probability of the given model being selected for the next population depends on the fitness value. The higher fitness value means higher chance of survival and generates more copies. In this way GA searches for new generations with gradually improved behavior using three fundamental operators. Reproduction, the strings with larger fitness values, produces large number of their copies in the new generation. In this way, fitter solutions have a higher chance to survive while weaker ones perish.

There are many different ways for the reproduction in the literature. One of the simplest procedures is the roulette wheel selection scheme. Other methods for selection are proportionate selection and tournament selection. By crossover the strings can exchange information probabilistically with each other. For this aim, each pair is selected progressively and a random number between 0 and 1 is generated. This number is compared to a crossover probability, pc. If the random number is greater than pc, then the two parents pass into the next generation unchanged. If not, then the parents are crossed over. There are three types of crossover: single point, multi-point and uniform crossover. By mutation, the strings can change their structure at randomly selected bit position. The bits of a string are independently mutated. Mutation may generate the strings which might not be produced by reproduction and crossover. So, this process is complementary to other operators. In this process, all bit positions are tested for mutation by generating a random number and comparing it with the mutation probability, pm. If it is less than pm, the bit is changed (1 to 0 or 0 to 1). Otherwise it is unchanged. If pm_/1, all bits may undergo mutation. There are no general termination criteria for GA. Predetermined number of generation or time or comparison of the best solutions to average fitness may be taken as stopping criterion.

Population size (N), crossover probability (pc) and mutation probability (pm) are known as the control parameters of GA. The values of these parameters must be specified before the execution of GA. These parameters depend on the nature of the objective function (Scott, M., T. DePauw, 2004; Durand, N., J.B. Gotteland, 2006; Cantoni, M., M. Marseguerra, 2000; Tomkins, G., F. Azadivar, 1995; Goldberg, EE., 1989). The structure of a Genetic Algorithm is shown in Figure 1.

The Integrated GA-Simulation:

In this paper we introduce a hybrid GA simulation approach. Moreover, we use the GA for calculating the fitness function. In fact each solution that exists in the GA consists of two numbers with the first number representing the green time of traffic light in line 1 and the second number represents the green time of traffic light in line 2. These numbers as parameters are given to the simulation problem as inputs, the simulation problem is run with these numbers and output is a number that equals to the average time of system per vehicle. In this paper the numbers of initial solutions are 10. After calculating the value of fitness function for initial solutions, we go to the next GA step. In this step we generate a number of solutions using GA operators: Mutation & Crossover that have been explained in the genetic algorithm section. In step 5 solutions are generated. Then, we use the initial solutions and the solutions that generated in the mutation and crossover steps to generate a new generation and substitute initial generation. Using the initial generation for generating a new generation ensures that the new generation is better than the initial generation. In fact we select 10 solutions from 20 generated solutions. In this problem 10 better solutions have been selected. A solution is better if its fitness is less than the others. Now, we consider the new generation like the initial generation and repeat the above steps and we repeat it until the stopping criteria is achieved. The stopping criteria can be one of the following items: 1) Reach to a desired value according to the framework of problem and the objectives of the problem. One of the weak points of this approach may be that the desired value isn't accessible in which the problem drops into an infinite loop. 2) The number of generated generations. In this paper the stopping criteria is achieved after 10 generations. The structure of the Hybrid GA Simulation is shown in Figure 2.

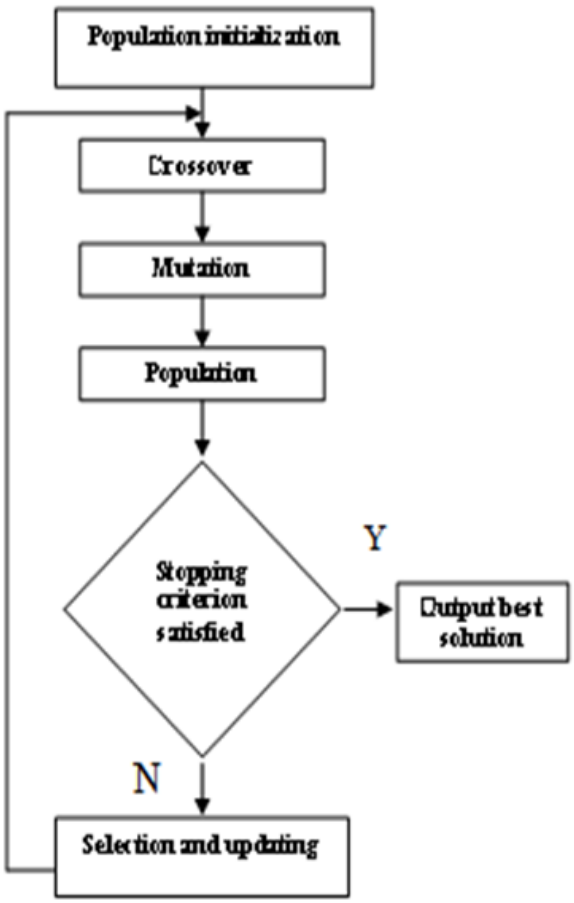

Fig. 1: The structure of a Genetic Algorithm

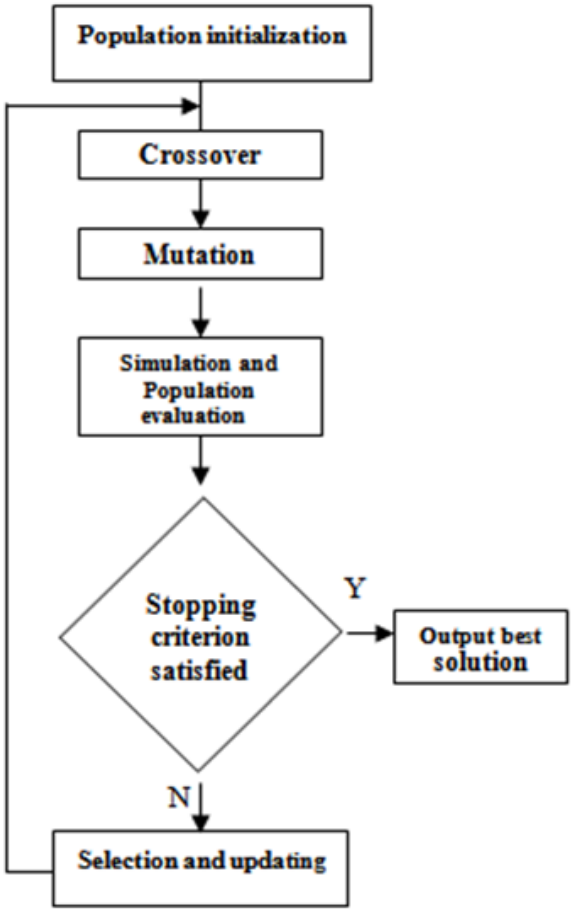

Fig. 2: The structure of the Hybrid GA Simulation

The Case Study:

The system used for simulation is a two line street that in which 500 meters is blocked for repair. The traffic lights are at the two ends of streets to control the traffic along of the road that is being repaired. Traffic lights in space of time allow traffic move only in one direction. When traffic light is green, vehicles that are waiting cross the light in 2 seconds. If a vehicle reaches green light and there is no queue, the vehicles cross the traffic light without any delay. Vehicles enter exponentially with an average of 12 and 9 seconds average for lines 1 and 2, respectively. The traffic light cycle consists of: green in line 1, both red, green in line 2, both red and repeat. The duration of red light is 55 seconds. The aim of the simulation analysis is to appoint sufficient time for time that traffic light is green in each line. This is continued until the wait time is minimized. Visual SLAM was used to model and simulate the traffic system. In this network variables are defined as follows:

XX (Chou, C.Y., C.H. Chen, 2001): the time that the light is green in line 1
XX [2]: the time that the light is green in line 2
XX [3]: number of observations in line 1
XX [4]: total time that vehicles are in system in line 1
XX [5]: number of observations in line 2
XX [6]: total time that vehicles are in system in line 2

Schematic model of this problem is shown in Figure 3. The reader should refer to Appendix II for complete GA codes. The Visual SLAM network of the Integrated GA-Simulation is shown in Figure 4.

We have run the traffic control problem 25 times with Integrated GA-Simulation and conventional simulation approaches. The results are summarized in Table 1.

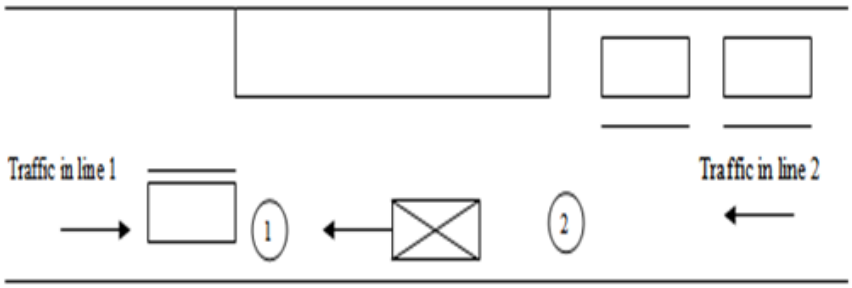

Fig. 3: The schematic model of the traffic system

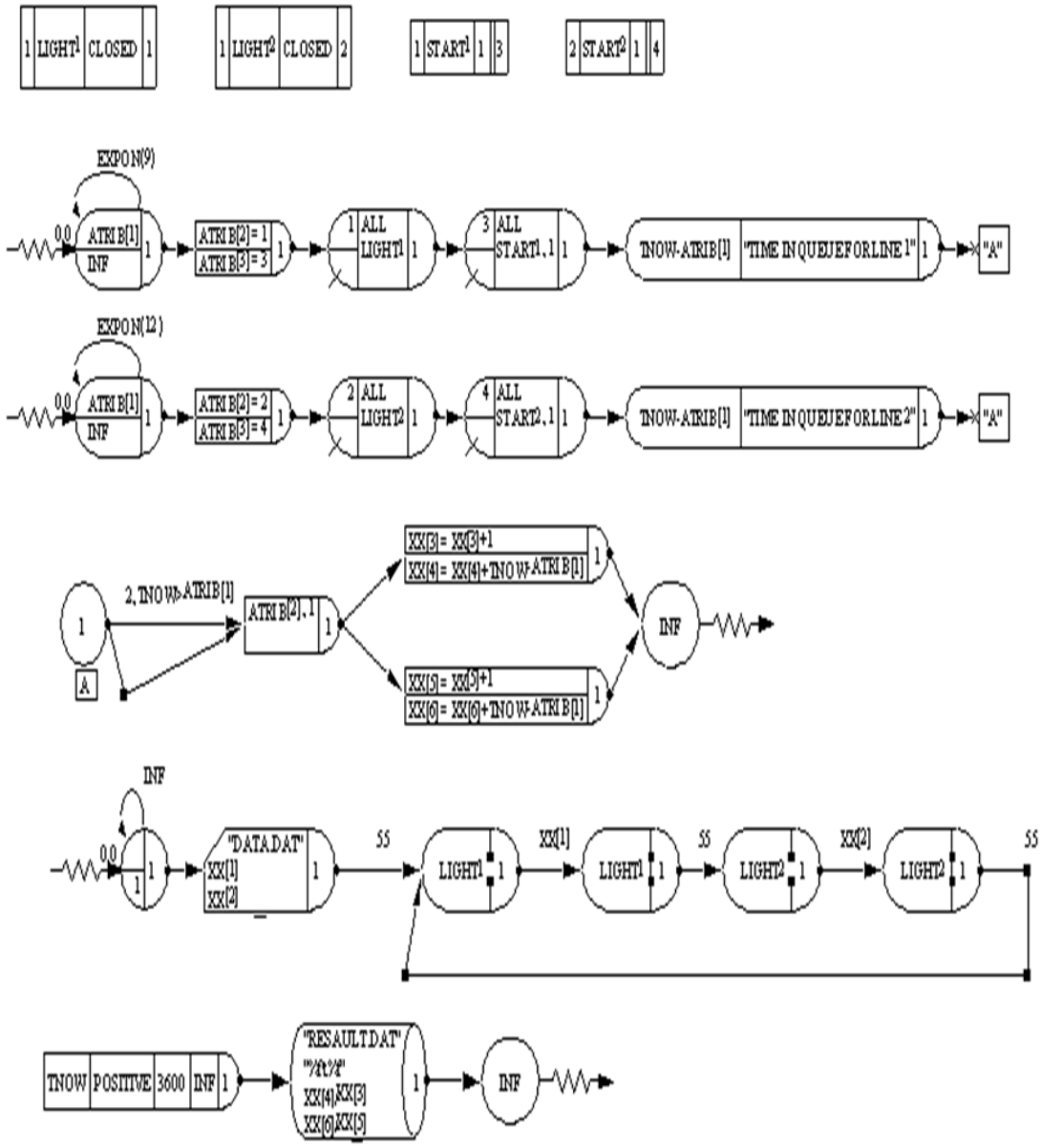

Fig. 4: The Visual SLAM network of the integrated GA simulation

Table 1: The average wait time of the Integrated-GA Simulation versus conventional simulation for 25 runs

SIMULATIONRUN	CONVENTION	HYBRID GA SIMULATION
1	77.6	70.098
2	78.45	70.221
3	77.614	72.456
4	76.068	69.825
5	78.264	70.222
6	78.026	70.482
7	77.108	69.675
8	77.832	68.682
9	77.537	71.663
10	75.126	69.615
11	77.471	70.405
12	77.705	70.041
13	77.99	78.353
14	71.554	69.981
15	78.53	69.705
16	75.725	70.228
17	82.275	69.158
18	75.458	71.075
19	76.907	70.844
20	79.212	69.416
21	74.476	71.082
22	78.597	71.775
23	78.09	68.451
24	74.074	70.998
25	76.563	70.832
average	70.611	77.13

At first, the equality of variances $H_0 : \delta_1^2 = \delta_2^2$ by F-test was tested. Hence, H_0 was accepted at α=0.01. Then, the null hypothesis $H_0 : \mu_1 = \mu_2$ was tested at α=0.01. We conclude that the average waiting time in the integrated GA-Simulation is less than that of conventional simulation. (T-statistic = 18.17 with 48 degrees of freedom and p-value = 0.000). The reader should note that subscripts 1 and 2 used for the above tests represent the two treatments or conventional simulation and Integrated GA-Simulation, respectively. The average waiting time when solved by GA simulation is 70.611 seconds whereas the average waiting time when solved by conventional simulation is 77.130 seconds. Therefore, the Integrated GA-Simulation improves the waiting time by 9.1%.

Conclusion:

In this study a problem related to traffic control systems is studied, analyzed and improved by the proposed Integrated GA-Simulation approach. Using GA in simulation is a strong tool for control, optimization and system performance upgrading. The waiting times for green lights depend on the rate of traffic. First, computer simulation is used to model the traffic control system. Then, simulation is integrated with GA to evaluate waiting times and time in system in the two directions. The average waiting time per vehicle and consequently the time in system are considerably lower than conventional simulation approach. This is shown through independent t-test. In this paper by combining simulation and GA the time that the light is green in lines 1 and 2 are 14 and 19 seconds, respectively. Moreover, the time in system decreased from 77.402 to 70.339 seconds. The average waiting time per vehicle is expected to have a reduction of about 9.1%. We can use the integrated model for design of an intelligent traffic light whose status changes during the day. This is the first study that introduces an integrated GA-Simulation approach for improvement of traffic systems with closed section for repair activity.

REFERENCES

Ayeni, J.O.A., 1990. Simulation model for timing road junction traffic lights, Modeling, Simulation & Control, 20(3): 53-64.

Cameron, D.B. Gordon, Duncan, I.D. Gordon, 1996. Parallel microscopic simulation of road traffic, Journal of Supercomputing, 10(1): 25-53.

Cantoni, M., M. Marseguerra, E. Zio, 2000. Genetic algorithms and Monte Carlo simulation for optimal plant design, Elsevier, Reliability Engineering and System Safety, 68: 29-38.

Chou, C.Y., C.H. Chen, M.H.C. Li, 2001. Application of computer simulation to the design of a traffic signal timer, Computers & Industrial Engineering, 39: 81-94.

De Schutter, B., B. De Moor, 1998. Optimal trac light control for a single intersection, European Journal of Control, 4(3): 260-276.

Durand, N., J.B. Gotteland, 2006. Genetic Algorithms Applied to Air Traffic Management, Part III Springer Berlin Heidelberg.

Goldberg, EE., 1989. Genetic algorithm in searching, optimization and machine learning. Addison-Wesley.

GiYoung, L., K. Jeong Jin, H. YouSik, 2002. The optimization of traffic signal light using artificial intelligence, IEEE International Conference on Fuzzy Systems, 3: 1279-1282.

Hesselink, H.H., H. Kuiper, J.M. van den Akker, 1996. Application of genetic algorithms in the aerospace domain EUFIT '96, the Fourth European Congress on Intelligent Techniques and Soft Computing, Aachen, Germany, September 2-5.

Huang, Y.S., T.H. Chung, 2009. Modeling and Analysis Urban Traffic Light Control Systems, Journal of the Chinese Institute of Engineers, 32(1).

Jianguo, Y., D. Wen, W. Jinmei, L. Qingfeng, W. Zhaoan, 2006. Modeling pedestrians' road crossing behavior in traffic system micro-simulation in China , Transportation Research Part A: Policy and Practice, 40: 280-290.

Jungblut-Hessel, R., B. Plateau, W.J. Stewart, B. Ycart, 2001. Fast simulation for road traffic network, RAIRO Recherche Operationnelle, 35(2): 229-250.

Kasun, H.N., Y.R. Janaka, 2004. Optimization of traffic signal light timing using simulation, Proc. Winter Simulation Conference, 2: 1428-1433.

Kun, C., Y. Lei, Z. Hui, 2008. Dynamic traffic simulation for accessing road network reliability under large-scale events: System development and case studies, Journal of System Simulation, 20: 5725-5730.

Oussedik, S. D.Delahaye, M. Schoenauer, 1999. Dynamic Air Traffic Planning by Genetic Algorithms. Proc. of the 1999 Congress on. Evolutionary Computation, CEC 99.

Pritsker, A.A.B., J.J. O'Reilly and D.K. LaVal, 1997. Simulation with Visual SLAM and AWESIM. New York, John Wiley.

Sadoun, B., 2008. On the simulation of traffic signals operation, Simulation, 84(6): 285-295.

Sanchez, J.J., E. Rubio, 2004. Genetic algorithms and cellular automata: a new architecture for traffic light cycles optimization, Proc. of the 2004 Congress on Evolutionary Computation(CEC) , Congress on Evolutionary Computation(CEC2004), 2: 1668-1674.

Scott, M., T. DePauw, 2004. AN Introduction to Genetic Algorithms, Journal of Computing Sciences in Colleges, 20(1): 115-123.

Tapani, A., 2005. Versatile model for simulation of rural road traffic, conf . Traffic Flow Theory 2005, 1934: 169-178.

Tomkins, G., F. Azadivar, 1995. Genetic algorithms in optimizing simulated systems. Proc. of the Winter Simulation Conference, SCS, San Diego, CA, 757-762.

Wang, D., H. Wang, Y.H. Zhao, Y. Gao, 2005. Interdomain Traffic Control over Multiple Links Based on Genetic Algorithm, Proc. of 2005 International Conference on, 5: 3036-3041.

Wen, W., C.L. Yang, 2006. A dynamic and automatic traffic light control system for solving the road congestion problem WIT Transactions on the Built Environment. Proc. 12th International Conference on Urban Transport and the Environment in the 21st Century, 89: 307-316.

Zhao, X.H., P.Y. Cui, 2003. Journal of Harbin Institute of Technology, 10(4): 453-456.

Appendix I: Control statement for the network is as follows:

```
1 GEN,,"TRAFFIC LIGHTS",,1,YES,YES;
2 LIMITS, 6,,, 2;
3 INTLC,{{XX[1],19},{XX[2],14},{XX[3],0},{XX[4],0},{XX[5],0}, {XX[6],0}};
4 NETWORK, READ;
5 FIN;
```

Appendix II: GA codes**:**

```
#include<stdlib.h>
```

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
float A[10][2],ans_cross[5][2],ans_mutation[5][2];
int h[3]={9,10,5},n_mutation=5,n_crossover=5,n_original=10,repeat=25;
void read_data();
void crossover();
float cross1(float,float,int);
float cross2(float,float,int);
void mutation();
float mut(float,int);
void write_data();
void main(){
 read_data();
 crossover();
 mutation();
 write_data();
}

////////////////////////////////////////////////////////////////////////////////

void read_data(){
 int i,j;
 FILE *fp;
 fp=fopen("D:/PROJECTS/TRAFFIC/DATA.dat","r");
 for(i=0;i<=n_original-1;i++)
 for(j=0;j<=1;j++)
 fscanf(fp,"%f",&A[i][j]);
 fclose(fp);
}

////////////////////////////////////////////////////////////////////////////////

void crossover(){
 int i,j,z,r1,r2,r3,r4,r5;
 randomize();
 for(i=0;i<=n_crossover-1;i++){
 r1=random(10);
 do{
 r2=random (10);
 } while (r1==r2);
 For (z=0; z<=1; z++)
 ans_cross[i][z]=A[r1][z];
 r3=random(3)+1;
 for(j=1;j<=r3;j++){
 r4=random(2);
 r5=random(2);
 if(r5==0)
 ans_cross[i][r4]=cross1(A[r1][r4],A[r2][r4],h[r4]);
 else
 ans_cross[i][r4]=cross2(A[r1][r4],A[r2][r4],h[r4]);
 }
 }
}

////////////////////////////////////////////////////////////////////////////////
```

```
float cross1(float a,float b,int pp){
 float a1,a2,b1,b2,ans;
 int k1,l1;
 randomize();
 k1=random(pp-2)+1;
 a1=(long int)a%(long int)pow(2,k1);
 a2=(long int)a%(long int)pow(2,k1+1);
 b1=(long int)a%(long int)pow(2,k1);
 b2=(long int)a%(long int)pow(2,k1+1);
 l1=random(2);
 if(l1==0)
 ans=a-a1+b1;
 else
 ans=b+a2-b2;
 return ans;
}

////////////////////////////////////////////////////////////////////////////////

float cross2(float a,float b,int pp){
 float a1,a2,a3,a4,b1,b2,b3,b4,ans;
 int k1,k2,l1,l2,r;
 randomize();
 k1=random(pp-2)+1;
 do{
 k2=random(pp-2)+1;
 }while((k1==k2)||(abs(k2-k1)==1));
 if(k1<k2){
 l1=k1;
 l2=k2;
 }
 else{
 l1=k2;
 l2=k1;
 }
 r=random(2);
 a1=(long int)a%(long int)pow(2,l1);
 a2=(long int)a%(long int)pow(2,l1+1);
 a3=(long int)a%(long int)pow(2,l2);
 a4=(long int)a%(long int)pow(2,l2+1);
 b1=(long int)b%(long int)pow(2,l1);
 b2=(long int)b%(long int)pow(2,l1+1);
 b3=(long int)b%(long int)pow(2,l2);
 b4=(long int)b%(long int)pow(2,l2+1);
 if(r= =0)
 ans=a+a2-a3+b3-b2;
 else
 ans=b+b1-a1+a4-b4;
 return ans;
}

////////////////////////////////////////////////////////////////////////////////

void mutation(){
 int i,r,r1,r2,r3,r4,k,j;
 randomize();
```

```
 for(i=0;i<=n_mutation-1;i++){
 r=random(10);
 for(k=0;k<=1;k++)
 ans_mutation[i][k]=A[r][k];
 r1=random(2)+1;
 for(j=1;j<=r1;j++){
 r2=random(2);
 r3=random(h[r2])+1;
 for(k=1;k<r3;k++){
 r4=random(h[r2]);
 ans_mutation[i][r2]=mut(ans_mutation[i][r2],r4);
 }
 }
 }
}

////////////////////////////////////////////////////////////////////////////////

float mut(float a,int k){
 float a1,a2,b;
 a1=(long int)a%(long int)pow(2,k);
 a2=(long int)a%(long int)pow(2,k+1);
 b=(a2-a1)/pow(2,k);
 return (a+(1-2*b)*(pow(2,k)));
}

////////////////////////////////////////////////////////////////////////////////

void write_data(){
 int i,k;
 FILE *fp1;
 fp1=fopen("D:/PROJECTS/TRAFFIC/INPUT.dat","w");
 for(i=0;i<=n_original-1;i++)
 for(k=1;k<=repeat;k++)
 fprintf(fp1,"%f\t%f\t%f\n",A[i][0],A[i](Chou, C.Y., C.H. Chen, 2001));
 for(i=0;i<=n_crossover-1;i++)
 for(k=1;k<=tekrar;k++)
 fprintf(fp1,"%f\t%f\t%f\n",ans_cross[i][0],ans_cross[i](Chou, C.Y., C.H. Chen, 2001));
 for(i=0;i<=n_mutation-1;i++)
 for(k=1;k<=tekrar;k++)
 fprintf(fp1,"%f\t%f\t%f\n",ans_mutation[i][0],ans_mutation[i](Chou, C.Y., C.H. Chen, 2001));
}
////////////////////////////////////////
void main(){
float r[20][3]={0},v[20][3]={0},sum_25[20][3]={0},e[3]={0},p[20][3]={0},sum1=0,sum[3]={0};
float q,sum13=0,rnd,new_nasl[10][3]={0},d[3]={0},w[3]={0},sum_2[3]={0},n[20][3]={0};
int i,j,k,exit1,repeat=25;
FILE *fp,*fp1;
randomize();
fp=fopen("D:/PROJECTS/TRAFFIC/RESAULT.dat","r");
for(i=0;i<=19;i++)
 for(j=1;j<=repeat;j++)
 for(k=0;k<=2;k++){
 fscanf(fp,"%f",&q);
 sum_25[i][k]+=q;
 }
```

```
for(i=0;i<=19;i++)
 for(j=0;j<=2;j++)
 r[i][j]=sum_20[i][j]/20;
for(i=0;i<=19;i++)
 sum13+=cl[i];
 b[0]=cl[0]/sum13;
 b[19]=1;
for(i=1;i<=18;i++)
 b[i]=b[i-1]+cl[i]/sum13;
for(i=0;i<=9;i++){
 rnd=((float)random(10000)+1)/10000;
 k=0;
 exit1=1;
 do{
 if(rnd<=b[k]){
 for(j=0;j<=2;j++)
 new_nasl[i][j]=r[k][j];
 exit1=0;
 }
 else
 k++;
 }while(exit1);
}
fp1=fopen("D:/DATA.dat","w");
for(i=0;i<=9;i++)
 fprintf(fp1,"%f\t%f\t%f\n",new_nasl[i][0],new_nasl[i](Chou, C.Y., C.H. Chen, 2001),new_nasl[i][2]);
fclose;
```