MULTIPLE OBJECTIVE OPTIMIZATIONS IN EVOLUTIONARY COMPUTATION

Istvan Sztojanov, Ph.D., IEEE member and Sever Pasca, Ph.D.
Department of Applied Electronics and Information Technology,
University "Politechnica" of Bucharest, Romania
(szistvan@colel.pub.ro)

Abstract: – **Road traffic flows in a crossing road point are representing a highly unpredictable and dynamic environment. In order to get the traffic optimization it is necessary to design and implement an unconventional evolvable system able to adapt itself to this rapidly changing environment. A multiple objective optimization process based on genetic algorithm is proposed hereafter, in order to find optimal ON-time values of the green and red lights, on both directions and axis of a crossing road point.**

Index terms: **evolutionary algorithms, multiple objective optimizations, and adaptive systems.**

Introduction

In a search problem of the real world, many possible competing objectives must be satisfied simultaneously. This topic is known as *multiple objective optimization*. In this case, it is almost impossible to find a unique optimal solution; instead of this, a family of sub-optimal solutions will characterize the optimization problem.

In order to cope with the problem of multiple objective optimization, effective and efficient methods must be designed and implemented since the conventional ones (designed for one-dimensional optimization) are not really suitable in this case. Genetic algorithm (GA) based on evolutionary computation seems to be more suitable for multiple objective optimization since the search for optimal solutions could be more effective and efficient.

Evolutionary computation applied for optimization process

The evolutionary algorithms are based on the performance evaluation of the individuals by their fitness value. In the case of a simple objective optimization, for a particular individual x a scalar value $f(x)$, called fitness, is computed. For the multiple objective optimization, a fitness vector $F(x)$ must be defined as $F(x) = (f_1(x), f_2(x), \dots, f_n(x))$ where $f_i(x)$ represents a scalar component of $F(x)$ corresponding to the fitness of the individual x with regard to objective i. The optimization problem now is to find optimal values for all functions $f_i(x)$, which, in the case of a real problem, it is almost impossible, so that it is more realistic to look for and to find acceptable values for all established objectives.

To tackle the above-mentioned problem, some evolutionary computing techniques have been proposed:

Plain aggregating approaches

In this case, the objective vector is transformed in a scalar by a weighted sum:

$$F(x) = \sum_i w_i \cdot f_i(x)$$

The main advantage is the simplicity of this strategy, but there are some drawbacks regarding the difficulty of correct setting of the weights (w_i) and the absence of interaction with the decision maker. During the search process, very often, some objectives are over-satisfied and others are under-satisfied within the solutions proposed.

Population based non-Pareto approaches

A multiple-objective optimization according to this algorithm is based on the selection of sub-populations from the old generation according to the individual's performance regarding one particular objective. After completing of these selection processes, sub-populations are merged together through the genetic operators of crossover and mutation.

Energy minimization strategy

This strategy derives from the plain aggregating approach in the sense that the individual fitness $f(x)$ is defined as a weighted sum of the normalized values $f_{ni}(x)$ of the fitness associated to the objective i:

$$f(x) = \sum_i w_i \cdot f_{ni}(x)$$

The weights must increase proportionally to the inverse of the satisfaction level of an objective. Based on this requirement the following weights updating scheme is proposed:

$$w_{i,t+1} = k_1 \cdot \alpha \cdot w_{i,t} + k_2 \cdot (1-\alpha) \cdot e_{i,t}$$

where $w_{i,t+1}$ is the weight associated to the objective i in the cycle t of the evolutionary algorithm, $e_{i,t}$ is the system error to satisfy a particular objective i, α is a real parameter which balances the contribution of the two terms, k_1 and k_2 are special parameters depending on the cumulative value of weights and error [1]. This mechanism enables the proportionality of the cumulative value of the weights to the cumulative value of the error. For the evaluation of the effectiveness of this method, in the sense of convergence of the overall system, the following energy measurement is used:

$$E = \sum_i w_i^2$$

An evolutionary system will perform well if the energy of the system is decreasing and an obvious condition to this aim is the decrease of the error associated to each objective i.

Multiple objective optimization algorithm for a traffic light controller

Traffic optimization means to design a traffic light controller being able to adapt the red and green time intervals, on both directions, to the varying traffic parameters in order to maintain a minimum number of vehicles in the cross road area. In fig. 1 there are represented a crossing road point and the parameters used for its description.

The parameters have the following meanings:

- $N_{x1}, N_{x2}, N_{y1}, N_{y2}$ – amounts of waiting vehicles at the beginning of current processing cycle;
- $v_{ix1}, v_{ix2}, v_{iy1}, v_{iy2}$ – amounts of incoming vehicles;
- $v_{ox1}, v_{ox2}, v_{oy1}, v_{oy2}$ – amounts of outgoing vehicles when the light is green on the given direction.

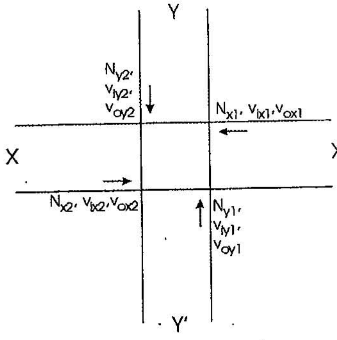

Fig. 1 – Crossing road point and parameters used for its description

Depending on the roads traffic, all these parameters are changing their values during 24 hours. If T_x and T_y are the time intervals for which the light is green for a given direction, amounts of waiting vehicles after a complete processing cycle $T = (T_x + T_y)$ are given by the following relations:

$$N_{x1}(T) = N_{x1} + v_{x1} \cdot (T_x + T_y) - v_{ox1} \cdot T_x \quad (1)$$

$$N_{x2}(T) = N_{x2} + v_{x2} \cdot (T_x + T_y) - v_{ox2} \cdot T_x \quad (2)$$

$$N_{y1}(T) = N_{y1} + v_{y1} \cdot (T_x + T_y) - v_{oy1} \cdot T_y \quad (3)$$

$$N_{y2}(T) = N_{y2} + v_{y2} \cdot (T_x + T_y) - v_{oy2} \cdot T_y \quad (4)$$

Thus, the total amount of waiting vehicles in the crossing road point, after a complete processing cycle $T = (T_x + T_y)$ is given by:

$$N(T) = N_{x1}(T) + N_{x2}(T) + N_{y1}(T) + N_{y2}(T) \quad (5)$$

A genetic algorithm based optimization of the total amount of vehicles in the crossing road point was presented in [3]. The drawback of the method is the fact that it could happen that in one direction the condition is over-satisfied and in the other direction it remains under-satisfied. For this reason, we propose a multiple objective optimization in order to obtain an acceptable minimum amount of waiting vehicles on every direction of both axis x and y.

For the multiple objective optimizations we define *the fitness vector* as it follows:

$$F(T) = (f_1(T), f_2(T), f_3(T), f_4(T)) \quad (6)$$

where, as individual objective fitness we use the quadrate value of the accumulated amount of waiting vehicles on a given direction.

To transform the fitness vector $F(x)$ into a scalar, a plain aggregating approach is used:

$$f(x) = \sum_i w_i \cdot f_i(x) \quad (7)$$

The simplest form of the relation (7) is obtained when we put $w_1 = w_2 = w_3 = w_4 = 1$, and the *fitness scalar* becomes

$$f(T) = N_{x1}^2(T) + N_{x2}^2(T) + N_{y1}^2(T) + N_{y2}^2(T) \tag{8}$$

The optimization model

To solve the above-mentioned optimization, a genetic algorithm was used in an off-line optimization process. T_x and T_y time intervals received eight values (15s, 30s, 45s, 60s, 75s, 90s, 105s, and 120s). Each of these was coded as a three-bit word resulting a six bit long string (Fig 2).

T_{x2}	T_{x1}	T_{x0}	T_{y2}	T_{y1}	T_{y0}

Fig. 2 – The binary string representing the possible solutions

From the 64 possible combinations, eight combinations were randomly chosen in order to form the initial population to be evolved. At the beginning of simulation, one string was randomly chosen in order to control the light controller. A standard GA was used with linear rank selection, one point crossover (with $p_c = 0.6$) and mutation (with $p_m = 0.01$). For the evaluation of the performance of the individuals from the given population the fitness value given by relation (8) was used. After the GA was applied, from the evolved population the best-fitted individual was chosen for giving T_x and T_y to control the light controller for the next period of time.

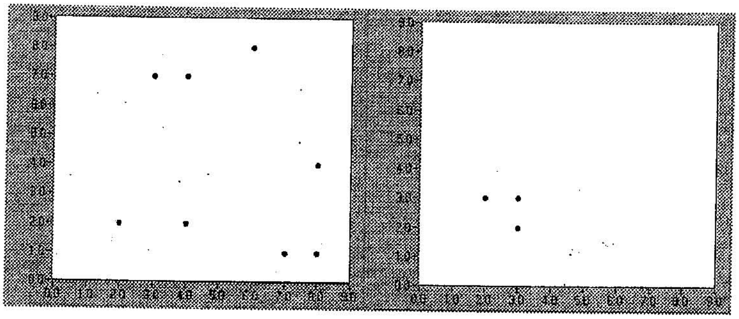

Fig. 3 – Distribution of original (left) and final (right) population

Results

Within the fig. 3 there are presented the distribution of the original, randomly chosen population (left) and the final distribution after 75 generations (right). One could remark the concentration of the individuals in a small, best fitted region of the search space and the fact that super individuals dominate the selection process.

These systems accomplish the optimization process by reducing the scalar quantity of the fitness function, as it is presented in fig. 4.

Within fig. 5 it is represented the 24 hours evolution of the amount of vehicles in the crossing road point with a deterministic control of the traffic. One could remark that on both directions of axis x (i.e. x1 and x2), between 12:00 and 16:00, the traffic is nearly blocked by excessive amounts of accumulated vehicles.

Within fig. 6 it is represented the 24 hours evolution of the amount of vehicles in the crossing road point with an evolutionary control of the traffic. We can remark that the evolutionary control avoids the blocking tendency of the traffic on both directions of axis x (i.e. x1 and x2).

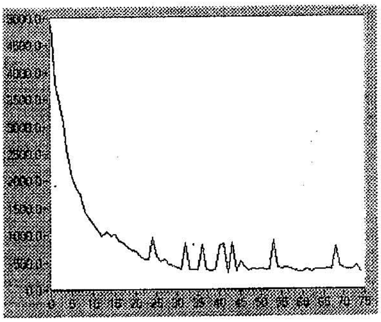

Fig. 4 – The average value of scalar fitness function during evolution

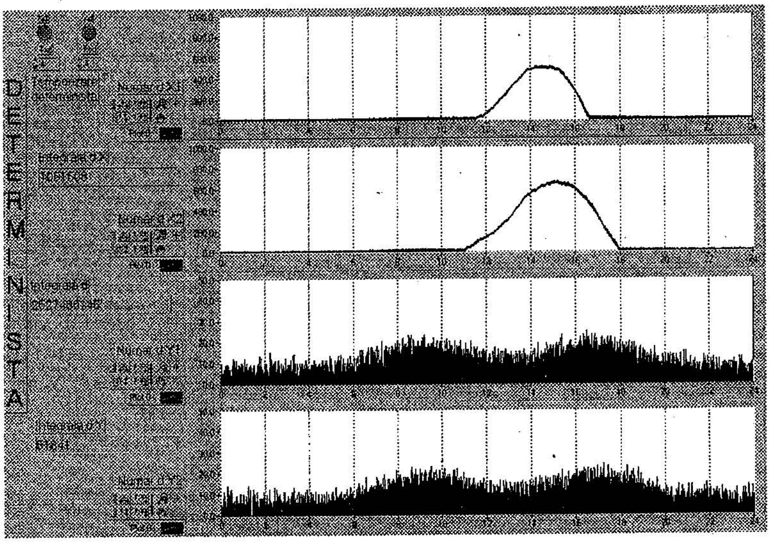

Fig. 5 – Timely evolution of vehicles amount in the crossing road point with a deterministic control of the traffic

One could consider that the evolutionary system is performing well if the accumulated amount of vehicles is decreasing. It is obvious that **the evolutionary approach, based on multiple objective optimization, is much more effective and efficient as the standard deterministic one**. In order to prove this, in the table 1 there are shown the total amount of vehicles waiting during 24 hours for both deterministic and evolutionary traffic control methods.

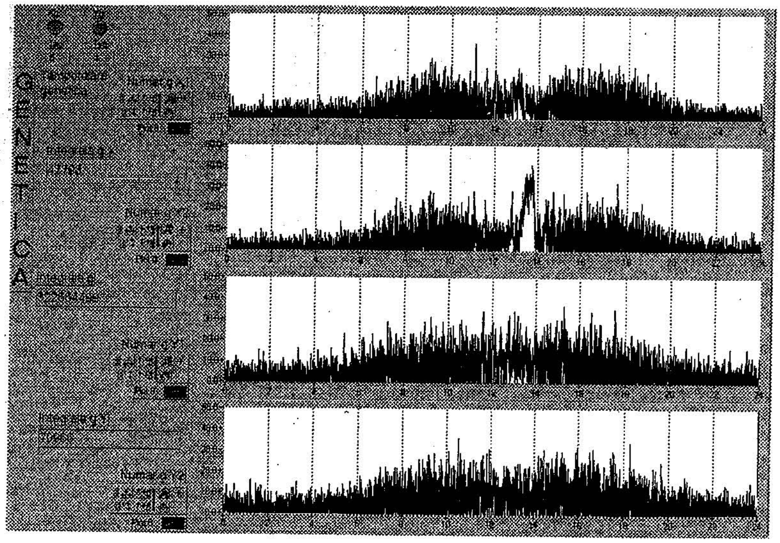

Fig. 6 – Evolution of the amount of vehicles in the crossing road point with the proposed evolutionary control of the traffic

Table 1 – Total amount of vehicles waiting during 24 hours

Type of traffic control	On x axis	On y axis	On both directions
Deterministic method	1.051.508	51.841	2.527.480.145
Evolutionary method	47.793	70.966	322.664.499

Conclusions

Results confirm the fact that **genetic algorithm based optimization is a promising method to solve problems under ill-defined conditions**. The proposed multiple objective optimization enables a minimum amount of vehicles blocked in the crossing area, through uniform distribution of the vehicles on every direction of both axis x and y of a road crossing point.

References

[1] R.S. Zebulum, M.A. Pacheco, M.M. Velasco, "*Evolutionary Electronic – Automatic Design of Electronic Circuits and Systems by Genetic Algorithms*", CRC – PRESS, Boca Raton, Florida, 2002.

[2] E. Sanchez, M. Tomassini, "*Towards evolvable hardware*", Springer Verlag, Berlin Heidelberg New York, 1996.

[3] I. Sztojanov, S. Paşca, "*On a genetic Algorithm based optimization study for a traffic light controller*", The 10-th International Scientific and Applied Science Conference – ELECTRONICS ET 2001, September 26–28, Sozopol, Bulgaria, 2001.