Published conference paper
Koh, Andrew (2009) *An adaptive differential evolution algorithm applied to highway network capacity optimization.* In: Avineri, E., Köppen, M., Dahal, K., Sunitiyoso, Y. and Roy, R., (eds.) Applications of Soft Computing: Updating the State of the Art. 12th On-line World Conference on Soft Computing in Industrial Applications, October 16 to 26 2007, Cyberspace. Advances in Intelligent and Soft Computing (52). Springer , Berlin.

http://dx.doi.org/10.1007/978-3-540-88079-0_21

An Adaptive Differential Evolution Algorithm Applied to Highway Network Capacity Optimization

Andrew Koh[1]

[1] Institute for Transport Studies, University of Leeds, 39 University Road, Leeds LS2 9JT, United Kingdom
{a.koh@its.leeds.ac.uk}

Abstract. Differential Evolution (DE) is a simple heuristic for global search. However, it is sensitive to certain control parameters and may not perform well if these parameters are not adjusted to suit the problems being optimized. Recent research has reported on methods to endogenously tune these control parameters during the search process. In this work, we develop and apply two DE variants as solution algorithms for continuous network design problems and illustrate with examples from the highway transportation literature.

Keywords: Continuous Network Design Problem, Differential Evolution, Bi-Level Programming

1 Introduction

The continuous network design problem (CNDP) involves the determination of capacity enhancements, measured as continuous variables, of existing facilities of a network in such a way that the decision is regarded as optimal [1]. This remains a challenging research area within transportation [2].

In the literature, Genetic Algorithms (GA) [3] and Simulated Annealing (SA) [4] have been used to tackle the CNDP. Such stochastic search algorithms are capable of providing globally optimal solutions for many multi-modal optimization problems such as those encountered in the CNDP and its variants.

In this paper, we investigate two variants of Differential Evolution (DE) [5], a stochastic search heuristic, as solution algorithms for the CNDP. DE has already been used successfully in a multitude of applications to solve real world engineering problems. Like other stochastic search techniques, however, the performance of DE is susceptible to the choice of certain user defined control parameters. These control parameters are problem dependent; while one set of parameters may work well on some problems, they may not perform as well on others [6]. Thus, significant resources have to be devoted to adjusting these parameters. Hence many researchers have proposed adaptive DE algorithms where these control parameters themselves are evolved during the optimization process [6,7,8,9].

This paper is organized as follows. Following this introduction, Section 2 provides an overview of the CNDP. Section 3 then reviews the DE algorithm and a variant that allows endogenous parameter adaptation. Section 4 illustrates the performance of DE on two CNDPs from the highway transportation literature and finally Section 5 offers some conclusions and directions for further research.

2 Continuous Network Design Problem (CNDP)

The CNDP can be categorized as a Mathematical Program with Equilibrium Constraints ("bi-level programs"). These are mathematical programming problems with an equilibrium condition inextricably embedded in its formulation. Such programs are equivalent to Stackleberg or leader-follower games in which the leader chooses her decision variables so as to optimize her objective, taking into account the response of the followers who optimize their separate objectives [10].

In this game, the leader is the network planner/regulator and the followers are the network users. The users treat the planner's decision on capacity as exogenous when deciding their route choice. The usual assumption in the CNDP is that the route choice for a given level of capacity enhancement is based on Wardrop's user equilibrium principle [11] where user equilibrium is attained when no user can decrease his travel costs by unilaterally changing routes.

The difficulty with the bi-level program is that the leader cannot optimize her objective without considering the reactions of the followers. Even when both the leader's problem and the follower's problem separately consist of convex programming problems, the resulting bi-level problem itself may be non-convex [2]. Non convexity suggests the possibility of multiple local optima. Furthermore, additional capacity can counter-productively increase the total network travel time; this is the well-known Braess's Paradox [12]. Hence attention must be paid also to the network effects of providing additional link capacity which road users do not consider in their route choice decisions.

2.1 Model Formulation

Let:

$N:(X,A)$ represent the transportation network with X nodes and A links

R : the set of all routes in the network

H : the set of all Origin Destination (OD) pairs in the network

R_h : the set of routes between OD pair $h\,(h \in H)$

D_h : the demand between each OD pair $h\,(h \in H)$

f_r : the flow on route $r\,(r \in R)$

v : the vector of link flows, $v=[v_a]\ (a \in A)$

$t_a(v_a)$: the travel time on the link a, as a function of link flow v_a on that link only.

δ_{ar} : 1 if the route $r\,(r \in R)$ uses link $a\,(a \in A)$, 0 otherwise

K : the set of links that have their individual capacities enhanced. ($K \subseteq A$)

β : the vector of capacity enhancements $\beta=[\beta_a],\ a\,(a \in K)$

$\beta_a^{\max}$, $\beta_a^{\min}$: the upper and lower bounds of capacity enhancements $a\,(a \in K)$

d_a : the monetary cost of capacity increments per unit of enhancement $a\,(a \in K)$

C_a^0 : existing capacity of link a $(\forall a \in A)$

θ : conversion factor from monetary investment costs to travel times

The CNDP seeks a K dimension vector of capacity enhancements that is optimal to the following bi-level program:

The Upper level problem (Program U) is given by

$$\underset{\beta}{\text{Min}}\, U(v,\beta) = \sum_{\forall a \in A} v_a(\beta) t_a(v_a(\beta), \beta_a) + \sum_{\forall a \in K} \theta d_a \beta_a \quad \textbf{(1)}$$

Subject to

$$\beta_a^{\min} \le \beta_a \le \beta_a^{\max} \quad \forall a \in K \quad \textbf{(2)}$$

Where v is obtained by solving the following lower level problem (Program L):

$$\underset{v}{\text{Min}}\, L = \sum_{\forall a} \int_0^{v_a} (t_a(z, \beta_a))dz \quad \textbf{(3)}$$

Subject to

$$\sum_{r \in R_h} f_r = D_h \,, h \in H \quad \textbf{(4)}$$

$$v_a = \sum_{r \in R} f_r \delta_{ar} \,, \forall a \in A \quad \textbf{(5)}$$

$$f_r \ge 0 \,, \forall r \in R \quad \textbf{(6)}$$

Program U defines the decision maker's objective as the sum of network travel times and investment costs of link capacity enhancements while Program L determines the user equilibrium flow, for a given β , based on Wardrop's first principle of route choice [11], formulated as an equivalent minimisation problem [13]. With β fixed, Program L can be solved via a traffic assignment algorithm.

The CNDP has been investigated by many researchers and various solution algorithms have so far been proposed. These have included Augmented Lagragian (AL) marginal function method [14] and Karush Khun Tucker (KKT) approaches [15]; both of which are derivative based methods. An approximation to Program U for an assumed β can be derived and direct search (DS) heuristics (i.e. search techniques that do not require derivatives such as golden section search) iteratively applied to approximately solve the CNDP [16]. Stochastic optimization techniques have also been used; GAs were applied in [17] and the use of SA has been reported in [1].

3 Differential Evolution Based Algorithms for CNDP

Differential Evolution (DE) is a multi-population based search heuristic [5] and has already been applied to a variety of real-world problems [18-20].

Our solution method using DE is as follows: At each iteration ("generation" in DE parlance), DE manipulates a population that comprises vectors of capacity enhancements (β) which are used to solve Program L , to obtain the link flows (v) to evaluate $U(v,\beta)$ and determine the "fitness" of a given β . A fitter vector implies a lower value for $U(v,\beta)$ since we aim to minimize $U(v,\beta)$ (equation 2). This population is then transformed via DE operations (discussed herein) to create a new population with improved fitness and the entire process is repeated.

In this section, we outline two variants of DE. The first, which we refer to as "Basic DE" is the original DE version in [5] which operates with user specified control parameters. The second, which we refer to as "Adaptive DE", endogenously tunes these control parameters. Table 1 shows pseudo code of the two DE variants. The processes of initialization, evaluation, mutation, crossover and selection are common to both variants and discussed next.

Table 1. Comparison of Basic DE and Adaptive DE procedures in pseudo code

Procedure Basic DE
Generation = 1
Initialization
Evaluation
REPEAT
    Mutation
    Crossover
    Evaluation
    Selection
UNTIL(Generation = MaxG)
	Procedure Adaptive DE
	Generation = 1
	Initialization
	Evaluation
	REPEAT
	 Create Control Parameters
	 Mutation
	 Crossover
	 Evaluation
	 Selection
	 Update Control Parameters
	UNTIL (Generation = MaxG)

Initialization

An initial population of size= NP capacity enhancement vectors, known as the parent population in DE parlance, is randomly generated using equation 7 as follows:

$$\beta_{i,a,G} = rnd(\beta_a^{\max} - \beta_a^{\min}) + \beta_a^{\min}, \forall i \in \{1,2,...,NP\}, \forall a \in K. \tag{7}$$

rnd is a pseudo-random number $\in [0,1]$.

Evaluation

Each member of the population is a K dimension vector of capacity enhancements (β). The evaluation process involves solving Program L to determine the resulting link flows and enables evaluation of Program U for each member of this population;

the member that results in the lowest objective function value for $U(v_a, \beta_a)$ is denoted the "best member" of the population ($\beta_{a,G}^{Best}$) at generation G.

Mutation

The mutation process combines different elements of the parent population heuristically to generate a mutant vector ($m_{i,a,G}$) in accordance with equation 8:

$$m_{i,a,G} = \beta_{i,a,G} + Q_i(\beta_{a,G}^{Best} - \beta_{i,a,G}) + Q_i(\beta_{a,G}^{r1} - \beta_{a,G}^{r2}) \quad \textbf{(8)}$$

$$\forall i \in \{1, 2, ..., NP\}, \forall a \in K$$

$r1, r2 \in \{1, 2, ..., NP\}$ are random integer and mutually different indices and also different from the current running index i. Q_i is a mutation factor that scales the impact of the differential variation. The mutation strategy shown in equation 8 is one of several variants proposed in [5].

Crossover

On this mutant vector ($m_{i,a,G}$) crossover is probabilistically performed to produce a child vector ($y_{i,a,G}$) according to equation 9 as follows:

$$y_{i,a,G} = \begin{cases} m_{i,a,G} & \text{if } rnd \in [0,1] < CR_i \ \vee \ i = h \\ \beta_{i,a,G} & \text{otherwise} \end{cases} \quad \forall i \in \{1, 2, ..., NP\}, \forall a \in K \quad \textbf{(9)}$$

$h \in \{1, 2, ..., K\}$: a random integer parameter index chosen to ensure that the child vector $y_{i,a,G}$ will differ from its parent by at least one parameter. CR_i is the probability of crossover.

Crossover can produce child vectors that do not satisfy bound constraints in equation 2. Out of bound values can be reset to a point half way between its pre-mutation value and the bound violated using equation 10 as suggested in [21].

$$y_{i,a,G} = \begin{cases} \dfrac{\beta_{i,a,G} + \beta_a^{\min}}{2} & \text{if } y_{i,a,G} < \beta_a^{\min} \\ \dfrac{\beta_{i,a,G} + \beta_a^{\max}}{2} & \text{if } y_{i,a,G} > \beta_a^{\max} \\ y_{i,a,G} & \text{otherwise} \end{cases} \quad \textbf{(10)}$$

Selection

Each child vector is compared against the parent vector. This means that comparison is against the same i^{th} ($\forall i \in \{1, 2, ..., NP\}$) vector parent on the basis of whichever of the two gives a lower value for Program U. Using this selection procedure also prevents the occurrence of Braess's Paradox in using the DE based algorithm since only when the objective of Program U is reduced would it be regarded as fitter. The one that produces a lower value survives to become a parent in the next generation as shown in Equation 11.

$$\beta_{i,G+1} = \begin{cases} y_{i,G} & \text{if } U(v_a(y_{i,G}), y_{i,G}) < U(v_a(\beta_{i,G}), \beta_{i,G}) \\ \beta_{i,G} & \text{otherwise} \end{cases} \quad \textbf{(11)}$$

$$S_{i,G} \ : 1 \text{ if } \beta_{i,G+1} = y_{i,G} \ ,0 \text{ otherwise} \quad \textbf{(12)}$$

For basic DE the procedures continue until some pre-specified number of generations ($MaxG$) are over. In Equation 12 $S_{i,G}$ is a dummy that takes on the value of 1 if the child vector survives or 0 otherwise. Equation 12 provides the link to Adaptive DE as will be shown next.

3.1 Adaptive DE

In basic DE it is conventionally further assumed [5] that:

$$Q_i = Q \ \forall i \in \{1,2,...,NP\} \quad and \quad CR_i = CR \ \forall i \in \{1,2,...,NP\} \quad \textbf{(13)}$$

In other words, in basic DE, it is assumed that the mutation and crossover factors are scalars. Thus basic DE requires 4 user-specified control parameters viz, Q, CR, NP and $MaxG$. There are some suggested values for these parameters. For example values such as $Q = 0.5, CR = 0.9$ have been suggested in [5,21]. However, in practice, many trial runs are required to find optimal parameters for each problem setting. Research developing adaptive versions tend to direct efforts at ways to endogenously compute Q and CR, leaving NP and $MaxG$ to remain user defined [6,7,8,9]. The latest contribution to a growing literature on parameter adaptation is in [9] which proposed the adaptive variant we discuss here. Other Adaptive DE versions can be found in [6,7,8].

The first point of departure for adaptive DE is to dispense with the assumption in Equation 13. This is done by separately associating each member of the population with its own crossover probability and mutation factor. Then, as an extension of equation 12, we may define the following:

$$S_{Q,G} = Q_i \cup S_{Q,G} \ \text{ if } S_{i,G} = 1 \ , S_{Q,G} = S_{Q,G} \ \text{ otherwise.} \quad \textbf{(14)}$$

$$S_{CR,G} = CR_i \cup S_{CR,G} \ \text{ if } S_{i,G} = 1 \ , \ S_{CR,G} = S_{CR,G} \ \text{ otherwise} \quad \textbf{(15)}$$

In other words, $S_{Q,G}$ and $S_{CR,G}$ denote the set of successful mutation factors and crossover probabilities used at generation G. Let the mean of these sets be denoted $\mu_{Q,G}$ and $\mu_{CR,G}$ respectively. Adaptive DE differs from Basic DE only in the creation and adaptation of the control parameters; steps shown in italics in the right pane of Table 1. All other processes are the same in both variants. The question then is how to obtain the vector of mutation and crossover factors for use at each generation in equations 8 and 9. We describe the methodologies to do so next.

Create Control Parameters

In addition to the initialization of a population of capacities, the user also specifies initial values of $\mu_{Q,G}$ = 0.7 and $\mu_{CR,G}$ = 0.5 (G=1). Then production of the control parameters were suggested as follows [9]:

- generating CR_i $\forall i \in \{1,2,...,NP\}$ randomly from a normal distribution with mean $\mu_{CR,G}$ and standard deviation 0.1 truncated to be real numbers between 0 and 1;
- generating one third of Q_i $\forall i \in \{1,2,...,\frac{1}{3}NP\}$ randomly from a rectangular distribution as real numbers between 0 and 1.2;
- and generating two thirds of Q_i $\forall i \in \{\frac{1}{3}NP+1, \frac{1}{3}NP+2,...,NP\}$ randomly from a normal distribution with mean $\mu_{Q,G}$ and standard deviation 0.1 truncated to be real numbers between 0 and 1.2.

Update Control Parameters

Following mutation, crossover and selection, the following steps are carried out to update $\mu_{Q,G}$ and $\mu_{CR,G}$ for the next generation

1. Compute the Lehmer mean (Lh) of $S_{Q,G}$ using equation 16.

$$Lh = \frac{\sum_{Q \in SQ,G} Q^2}{\sum_{Q \in SQ,G} Q} \quad \textbf{(16)}$$

2. Update $\mu_{Q,G+1}$ using equation 17.

$$\mu_{Q,G+1} = (1-c)\mu_{Q,G} + cLh \text{, where } c \in (0,1) \text{ is a user specified constant} \quad \textbf{(17)}$$

3. Compute Arithmetic mean ($\overline{S_{CR}}$) of $S_{CR,G}$
4. Update $\mu_{CR,G+1}$ using equation 18 as follows:

$$\mu_{CR,G+1} = (1-c)\mu_{CR,G} + c\overline{S_{CR}} \text{ where } c \text{ is as used in 17} \quad \textbf{(18)}$$

Once the means are updated, the generation method can be applied to create CR_i and Q_i parameters for the mutation and crossover processes. In summary, the main difference between adaptive DE and basic DE is the use of vector based control parameters; the parametric distributions used to generate these are iteratively adapted in the algorithm depending on the occurrence of successful selection.

4 Numerical Examples

We report on the performance of basic DE and adaptive DE based algorithms for solving the CNDP with other reported solutions on two test problems. In addition, we used the Origin Based Assignment algorithm [22] to solve the Program L during the evaluation phase of DE. Our reported results for the DE methods are the average upper level problem objective values i.e. $U(v,\beta)$ and its standard deviation (SD) over 30 runs. The results for other algorithms are taken directly from the cited sources.

4.1 Example 1

The CNDP for the hypothetical network of 16 links and 2 OD pairs is used as the first example. This network, its parameters and trip details are taken from [16]. All 16 links were subject to capacity enhancements with $\beta_a^{\min} = 0, \beta_a^{\max} = 20$ (for all links) and $\theta = 1$. For basic DE, we assumed Q and CR to be 0.8 and 0.95 respectively. For adaptive DE, we assumed c to be 0.01. In both cases $NP = 20$ and $MaxG = 150$. The results are shown in Table 2. Note that the gradient based methods [14, 15] can be mathematically shown to converge at least to a local optimum while both of our DE based methods are heuristics. As the CNDP is a non-convex problem [2], the ability of these former methods to locate the global optimum is dependent on the starting point assumed.

Table 2. Comparison of DE variants with other approaches for Example 1.

Method	KKT	AL	SA	GA	Basic DE	Adaptive DE
Source	[15]	[14]	[1]	[17]		
Objv	534.02	532.71	528.49	519.03	522.71	523.17
NEval	29	4,000	24,300	10,000	3,000	3,000
SD	------------Not Available----------			0.403	1.34	0.97

Note: Objv: value of $U(v,\beta)$ at end of run; NEval: number of Program L (traffic assignments) solved

4.2 Example 2

The second example is the CNDP for the Sioux Falls network with 24 nodes, 76 links and 552 OD pairs. The network and travel demand details are found in [16]. 10 links out of the 76 are subject to capacity enhancements; $\beta_a^{\min} = 0, \beta_a^{\max} = 25$ (for all the 10 links) and θ is 0.001. For basic DE, we assumed Q and CR to be 0.8 and 0.9 respectively. For adaptive DE, c was 0.01. In both cases $NP = 20$ and $MaxG = 80$. The literature does not indicate that GA has been used for this problem. The results are shown in Table 3.

Table 3. Comparison of DE variants with other approaches for Example 2.

Method	DS	KKT	AL	SA	Basic DE	Adaptive DE
Source	[16]	[15]	[14]	[1]		
Objv	83.08	82.57	81.75	80.87	80.74	80.74
NEval	12	10	2,000	3,900	1,600	1,600
SD	---------------Not Available--------------				0.002	0.006

Note: Objv: Value of $U(v,\beta)$ at end; NEval: number of Program L (traffic assignments) solved

Note that while this network is clearly larger and arguably more realistic, the problem dimension (number of variables optimized) is smaller than in Example 1, since 10 links are subject to improvement compared to 16 in Example 1. This could explain why the number of Program *L* problems solved are less than in Example 1.

5. Conclusions

We developed and applied a basic DE and adaptive variant as solution heuristics for the CNDP. In our numerical tests, we applied these methods to a hypothetical 16 link, 2 OD pair network and the Sioux Falls network with 76 links and 552 OD pairs.

As a global optimization heuristic, it has been concluded [5] that DE is competitive with SA and GA. Our results support this view as the DE-based methods required a lower number of function evaluations to obtain/better the optimum found by GA and SA. The GA based [17] results in example 1 are better but at the expense of extensive function evaluations. In future work, DE could be hybridized with local search algorithms to home in on the global optimum as in [6]. Our results for the Sioux Falls network in example 2 are arguably better than any in the literature so far.

Our results are also competitive against various derivative-based methods. Compared to the KKT methods in [15], DE required many more function evaluations but DE obtained a lower function value, suggesting DE managed to escape a local optimum. On the other hand, the gradient based AL method [14] required more function evaluations than the DE based methods but did not reach the optimum value.

The low variance augments the view that the DE methods are also reasonably robust. Hence this implies that the DE based method should be able to consistently locate the region of the optimum in multiple trials. Furthermore, a population size *NP* much less than suggested in [5] was used to obtain the results shown in this paper.

It is difficult to compare the performance of Basic and Adaptive DE since both could provide solutions that are quite similar. Furthermore, the standard deviations of both are not too different. Nevertheless, we point out that we carried out extensive initial testing to decide the mutation and crossover parameters ultimately used for the Basic DE algorithm which attests to the primary advantage of the adaptive version.

The downside of the adaptive variant is that further sensitivity analysis of the c parameter used for updating the crossover and mutation factors, needs to be carried out to examine its robustness with different test problems although [9] suggests that it is not critical. We conjecture that this parameter might be problem dependent. Further research is required to investigate this adaptive DE variant before firm conclusions can be reached.

Another potential avenue for further research could be to apply such stochastic search heuristics to multi-objective optimization CNDPs where tradeoffs between objectives need to be examined.

Acknowledgments. The work reported forms research funded by UK EPSRC. The author thanks Prof Hillel Bar-Gera (Ben Gurion University of the Negev, Israel) for provision of his Origin Based Assignment software. Comments on earlier drafts were received from Prof Mike Maher. Errors remain the author’s sole responsibility.

Reference

1. Friesz T., Cho H., Mehta N., Tobin R., Anandalingam G.: A Simulated Annealing Approach to the Network Design Problem with Variational Inequality Constraints Transport. Sci., 26, (1992), 18-26
2. Bell M., Iida Y.: Transportation Network Analysis, John Wiley, Chichester (1997)
3. Goldberg D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, Massachusetts (1989)
4. Aarts E., Korst J.: Simulated Annealing and Boltzman Machines,: John Wiley, Chichester (1988)
5. Storn R, Price K.: Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, J. Global Optim., 11,(1997), 341-359
6. Qin A., Suganthan P. N.: Self-adaptive differential evolution algorithm for numerical optimization, Proceedings of 2005 IEEE Congress on Evolutionary Computation, (2005), 1785-1791
7. Brest J, Bošković B., Grenier S., Žumer V., Maučec M. V.,:Performance comparison of self-adaptive and adaptive differential evolution algorithms, Soft Comput.,11,(2007),617-629.
8. Liu J., Lampinen J.: A Fuzzy Adaptive Differential Evolution Algorithm, Soft Comput.,9,(2005),448-462
9. Zhang J., Sanderson A.: JADE: Self-Adaptive Differential Evolution with Fast and Reliable Convergence Performance, Accepted manuscript submitted to 2007 IEEE Congress on Evolutionary Computation (2007). *Forthcoming*
10. Fisk C.: Game theory and transportation systems modeling Transport. Res. B-Meth., 18,(1984), 301-313
11. Wardrop J. G.: Some theoretical aspects of road traffic research, Proc. I. Civil Eng.-Tpt., 1(1952), 325-378
12. Braess D.: Uber ein paradoxon aus der verkehrsplanung, Unternehmenforschung 12 (1968), 258-268.
13. Beckmann M., McGuire C.B, Winsten C.B.: Studies in the Economics of Transportation, Yale University Press New Haven, Connecticut: (1956)
14. Meng Q., Yang H., Bell M.: An equivalent continuously differentiable model and a locally convergent algorithm for the continuous network design problem, Transport. Res. B-Meth,35,(2000),83-105
15. Chiou S.: Bilevel programming formulation for the continuous network design problem, Transport. Res. B-Meth. 39, (2005), 361-383
16. Suwansirikul C., Friesz T., Tobin R. L.: Equilibrium decomposed optimization, a heuristic for continuous network design, Transport. Sci., 21, (1997), 254-263
17. Cree N.D., Maher M.J., Paechter B.: The continuous equilibrium optimal network design problem: a genetic approach, in Bell, M. (ed.): Transportation Networks: Recent Methodological Advances, Pergamon, London, (1996), 163-174
18. Ilonen, J., Kamarainen J., Lampinen J.: Differential Evolution Training Algorithm for Feed-Forward Neural Networks, Neural Process. Lett., 7, (2003), 93-105.
19. Karaboga N.: Digitial IIR Filter Design Using Differential Evolution Algorithm, Eurasip J. Appl. Si. Pr., 8, (2005),1269-1276
20. Qing A.: Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems, IEEE Trans. Geosci. Remote Sens., 44,(2006),116-125
21. Price K.: An Introduction to Differential Evolution in Corne D., Dorigo M., Glover F. (eds): New Techniques in Optimization, McGraw Hill, London, (1999), 79-108
22. Bar-Gera H.: Origin based algorithm for the traffic assignment problem, Transport. Sci. 36,(2002),398-417