BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 59, No. 3, 2011
DOI: 10.2478/v10175-011-0032-z

A new genetic approach for transport network design and optimization

S. DINU* and G. BORDEA

Electronics, Electrotechnics and Informatics Department, Constanta Maritime University
104 Mircea cel Batran St., 900663 Constanta, Romania

Abstract. This paper presents an improved Genetic Algorithm to solve the Transportation Network Design Problem (CTNDP) with interactions among different links. The CTNDP is formulated in an optimal design as a bi-level programming model. A key factor in the present approach is the combination of diploid based complex-encoding with meiosis specific features. The novel mutation operator proposed is another improvement that leads to a better robustness and convergence stability.

The computational results obtained by comparing the performance of the proposed algorithm and other Genetic Algorithms for a test network demonstrates its better local searching ability, as well as its high efficiency.

Finally, suggestions for further research and extensions are given.

Key words: Genetic Algorithm, bi-level programming, Network Design Problem, complex-encoding.

1. The Transportation Network Design Problem – general description

The Transportation Network Design Problem (TNDP) involves optimal decisions in determining a set of design parameters for improving an existing transportation network in response to an increasing level of traffic demand. The general increase in flow level results in traffic congestion, delays, higher fuel and maintenance costs, air pollution and accidents.

The improvements of a transportation network, such as expansion of the capacities of the existing congested links, addition or deletion of links, traffic signal control adjustment, are made in accordance with a system optimum while considering the travel and route choice behavior of network users. The system optimum usually represents the minimization of the total travel time and construction costs.

The network user's decisions correspond to a set of nonlinear relations that are formulated as an independent mathematical programming problem.

In fact, a transportation network improvement involves the interaction of two parties with own objectives: the network planner represented by the transportation system authority and the network users that use the provided services. The traffic authority tries to optimize some overall objectives in the network, while the network users try to minimize their travel times/costs or perceived travel times/costs.

The decision variables of the network planner affect the route choice behavior of network users which is based on two equilibrium principles:

- the deterministic user equilibrium (DUE) condition [1] wherein network users choose the route with the shortest travel time/ the lower travel cost and equilibrium is reached where no user can unilaterally change routes to improve his/her own travel time or cost; assumptions of DUE can be somewhat unrealistic because of the variations in network conditions, variations in demand and no perfect information available for network users;
- the stochastic user equilibrium (SUE) condition [2] where no user can unilaterally change routes to improve his/her own perceived travel time or cost; SUE may reflect travelers' behavior more realistically than DUE. One can consider that DUE is a special case of SUE, when the variance of travel time perception is zero.

A mathematical programming model suitable for addressing this type of planning problem according to network economic criteria, saving both interests in an optimal design, is the bi-level programming model.

2. The transportation network design problem – bi-level optimization problem

Bi-level optimization is a useful approach for problems with conflicting objectives within a hierarchical structure.

It originated from the fields of game theory and decision making, also describes a number of problems in transportation planning and modeling (road network design, space localization, traffic control, optimal congestion pricing), engineering design (optimal structure and shape), economics (planning, envy-free pricing, etc). A detailed list of references and various proposed methods for solving bi-level programming applications is found in [3].

Its sequential framework involves two optimization problems at different levels, where the feasible set of the first problem, called the upper-level (leader) problem, is determined by

*e-mail: sedinu@yahoo.com

the other optimization problem, called the lower-level (follower) problem (Fig. 1). By extending this concept, one can define multi-level programs with any number of levels.

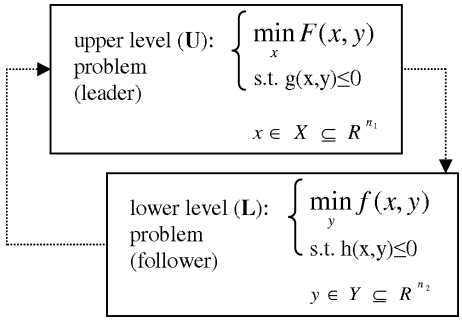

Fig. 1. Bi-level model

In Fig. 1:

- x and y are the decision variable vectors of the upper level and lower level, respectively;
- *F(x,y) and f(x,y)* are the leader's and the follower's objective functions, respectively;
- $g(x,y)$ and $h(x,y)$ are the leader's and the follower's constraints, respectively;
- X and Yrepresent upper and lower bounds on elements of the vectors x and y:

$$X = \{(x_1, \ldots, x_{n_1})^T \in \\ \in R^{n_1} | l_{s_j} \leq x_j \leq l_{d_j},\ j = 1, \ldots, n_1|\}, \tag{1}$$

$$Y = \{(y_1, \ldots, y_{n_2})^T \in \\ \in R^{n_2} | \bar{l}_{s_j} \leq x_i \leq \bar{l}_{d_j},\ i = 1, \ldots, n_2|\}. \tag{2}$$

For some particular cases of bi-level problems, the search spaces $X$and Y may introduce additional complicated requirements.

In Fig. 1 both the leader and the follower try to optimize their own objective function without considering the objective of the other one. The leader has complete information about the follower's optimization problem and its choice affects the lower level.

First, the leader makes a decision by selecting a vector $x \in X$ that optimizes its objective function. Then, for the given vector x, the follower reacts by selecting a vector $y \in Y$ that optimizes its own objective function. According to this, the leader integrates within its optimization process the feedback from the follower.

The TNDP is well studied and many alternative approaches can be found in the literature. Most of these approaches formulate the TNDP as a bi-level optimization problem, where the upper-level is to minimize the total system cost - the network planner's problem – and the lower level is the user equilibrium (UE) problem: deterministic UE or stochastic UE (Fig. 2).

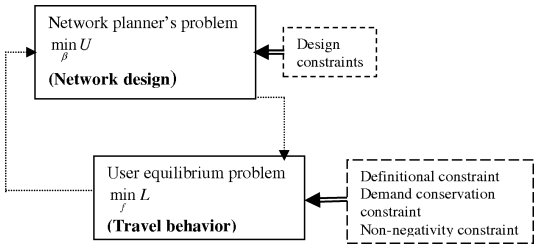

Fig. 2. TNDP- bi-level optimization

The upper-level problem can be formulated with different decision variables and objective functions. Based on the design variables of the upper-level problem, the resultant TNDP can also be categorized into:

- Discrete Transportation Network Design Problem (DTNDP): changes are made to the network's physical configuration and involve binary design variables such as addition of new links [4], the selection of the optimum configuration of one-way and two-way routes [5] or the bus network design problem [6].
- Continuous Transportation Network Design Problem (CTNDP): changes are made only to the attributes of the existing network and involves continuous design variables such as optimal capacity enhancement for a subset of existing links [7, 8], transit line frequencies [9], link tolls [10], etc.

Synthesizing the literature review, Chen and Alfa [4] took into consideration the shape of the objective function and further divided the problem into: TNDP with linear objective function, TNDP with nonlinear objective function of which the solutions satisfy the system-optimal criterion and TNDP with nonlinear objective function of which the solutions satisfy the user-optimal equilibrium criterion.

Both DTNDP and CTNDP defined above present increased computational complexity since the number of design variables is typically high and the set of constraints involves non-linear formulations. Also because external effects are to be considered.

Furthermore, 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. Non-convexity suggests the possibility of multiple local optima [11].

For solution approaches, several algorithms appropriate for addressing complex, combinatorial optimization problems have been proposed in the literature for solving both types of problems, formulated as a bi-level optimization model. The representative works involve: gradient-based methods and meta-heuristic techniques (Simulated Annealing – SA, Genetic Algorithms – GA, and Particle Swarm Optimization – PSO) to produce acceptable solution for large problems.

3. Model formulation

3.1. Notations. Starting from the basic design model which frequently occurs in the network literature, we develop an additional approach in order to make the CTNDP more realistic.

As it is a standard, the connected transportation network is represented by a graph $G = (N, A)$ defined by a set of nodes $n \in N$ and by a set of arcs $a \in A$, $|A| = k$.

O denotes the set of trip Origins, $o \in O$

D denotes the set of trip Destinations, $d \in D$

OD denotes the set of origin-destination pairs on the network, $(o, d) \in OD$

P denotes the complete set of available paths in the network

P^{od} denotes the set of paths in the network between $O-D$ pair *(o,d)*, $\forall (o, d) \in OD$

h denotes the vector of path flows, $h = [h_p^{od}]$, $\forall p \in P^{od}$, $\forall (o, d) \in OD$

Q denotes the vector of origin-destination demands, $Q = [q_{od}]$, $\forall (o, d) \in OD$
where

$$q_{od} = \sum_{p \in P^{od}} h_p^{od}, \tag{3}$$

Δ denotes the link-path incidence matrix, $\Delta = [\delta_{ap}^{od}]$, $\forall a \in A$, $\forall p \in P^{od}$, $\forall (o, d) \in OD$
where

$$\delta_{ap}^{od} = \begin{cases} 1, \text{ if link } a \in A \text{ is on path } p \in P^{od} \\ 0, \text{ otherwise} \end{cases}$$

Associated with each of the link $a \in A$ are the follow concepts:

f denotes the vector of link flows, $f = [f_a]$, $\forall a \in A$

f_a^s denotes the flow on link a
where

$$f_a = \sum_{(o,d) \in OD} \sum_{p \in P^{od}} \delta_{ap}^{od} h_p^{od}, \quad \forall a \in A, \tag{4}$$

β denotes the vector of link capacity enhancement, $\beta = [\beta_a]$, $\forall a \in A$ where β_a denotes the capacity enhancement of link $a \in A$, w denotes the vector of existing link capacity, $w = [w_a]$, $\forall a \in A$, where w_a denotes the current capacity of link $a \in A$.

Thus, the capacity after expansion will be $\beta_a + w_a$ $t(f, \beta)$ denotes the vector of link travel cost, $t(f, \beta) = [t_a(f_a, \beta_a)]$, $\forall a \in A$, where $t_a(f_a, \beta_a)$ denotes the travel cost on link $a \in A$.

The travel cost on a link increases as the flow increases because of traffic congestion. To allow for congestion, travel cost on a link is described as a function of link flow f_a and capacity enhancement β_a.

A common choice for the travel costs is the Bureau of Public Roads (BPR) function [12, 13]:

$$t_a(f_a, \beta_a) = t_a^0[1 + b_a(f_a/(\beta_a + w_a))^4]$$

where $t^0 \in \Re^A$ denotes the vector of free flow travel costs; t_a^0 denotes the free-flow travel cost on link a; b_a is the congestion parameter for link a and is calibrated on the basis of speed limit and the link capacity.

Most analytical models generally suppose that the travel time on a given link depends only on flow through that link (symmetric case). However, in a number of cases, link interactions could occur among different links.

Starting from [14] and [15] in this study we develop a travel cost function by modifying the BPR travel cost function:

$$t_a(f_a, \beta_a) =$$
$$= t_a^0 \left\{ 1 + b_a \left[(f_a + \sum_{b \in A} \varepsilon_{a,b} f_a)/(\beta_a + w_a) \right]^4 \right\}, \quad \forall a \in A, \tag{5}$$

where $0 \leq \varepsilon_{a,b} \leq 1$ denotes the "weighted impact factor" of the flow on link $b \in A$ to the travel cost of link a; $\varepsilon_{a,b} = 0$ when there is no interaction between link a and link b (the cost on link a is not dependent on the flow on link b); $\varepsilon_{a,b} = 1$ when $a = b$.

Otherwise, the values for $\varepsilon_{a,b}$ ranges from *0* to *1*.

The greater the influence which the flow on link b has on the travel cost on link a, the closer $\varepsilon_{a,b}$ is to the value of 1. The values for $\varepsilon_{a,b}$ are specific for a given network and are based on a one-year recorded data.

The matrix $E = (\varepsilon_{a,b}) \in \Re^A x \Re^A$ randomly generated in our case study represents the link interactions among different links. It is not symmetric since the cost on a link may depend on the flow on another link in a different way than the cost on the other link depends on the flow on that link. Such a generalization allows for a more realistic treatment of intersections, two-way links, multiple modes of transport as well as distinct classes of users of the network [11].

One can note that if $\varepsilon_{a,b} = 0$ the function formulation will reduce to the standard BPR travel cost function for the symmetric case.

C denotes the vector of path travel time, $C = [c_p^{od}]$, $\forall p \in P^{od}$, $\forall (o, d) \in OD$, where the travel time on a particular path is assumed to be a summation of the corresponding link travel times:

$$c_p^{od} = \sum_{a \in A} \delta_{ap}^{od} t_a(f_a, \beta_a), \tag{6}$$

where $g_a(\beta_a)$ denotes the investment cost function on link a; it gives the cost of increasing a's capacity by β_a. The investment cost function adopted in this study is:

$$g_a(\beta_a) = d_a \beta_a$$

where d_a represents the monetary cost of capacity increments per unit of enhancement and it is known, $\forall a \in A$.

In some test problems, quadratic investment cost functions allows to properly model the network improvement. For this case a suitable definition could be:

$$g_a(\beta_a) = d_a \beta_a^2.$$

The solution behavior under various investment cost formulations was investigated by Abdulaal and LeBlanc [7]. See also Marcotte and Marquis [16] for further details.

θ denotes a user defined factor converting investments costs to travel cost; in this study we set $\theta = 1$.

B is the total available budget for network capacity improvements.

3.2. Assumptions. Assumptions used in this paper for modeling the CTNDP are based on those made in [17]:

1) demand for each $O-D$ pair over some planning interval is fixed and it is known a priori; it is given in the form of an $O-D$ matrix.

2) the link travel function $t_a(f_a, \beta_a)$, $a \in A$ is strictly increasing and continuously differentiable with respect to link flow f_a, $a \in A$ for any fixed capacity enhancement β_a, $a \in A$.

3) the investment cost function $g_a(\beta_a)$, $a \in A$ is a continuous differentiable function with respect to β_a.

3.3. Formulation. The traditional objective of the Continuous Network Design Problem is to determine a set of expansion values β_a (decision variables) that minimizes total system cost (the sum of the total travel time cost and the investment cost of link capacity expansions):

$$\sum_{a \in A} [t_a(f_a(\beta), \beta_a) f_a(\beta) + \theta g_a(\beta_a)], \tag{7}$$

where the first term gives the travel time cost and the second term gives the government's total investment cost (capital construction costs, maintenance fees, etc.)

The flow on each link $a \in A$, f_a is the user equilibrium flow pattern and is obtained by solving the lower level problem.

The CTNDP under DUE based on Wardrop's first principle of route choice [1] can be formulated in terms of the bi-level programming model as follows:

Upper-level problem (U):

$$\min_{\beta} U(f, \beta) = \sum_{a \in A} [t_a(f_a(\beta), \beta_a) f_a(\beta) + \theta g_a(\beta_a)], \tag{8}$$

subject to:

$$\beta_a^{\min} \leq \beta_a \leq \beta_a^{\max}, \quad \forall a \in A, \tag{9}$$

$$\sum_{a \in A} g_a(\beta_a) \leq B. \tag{10}$$

Lower-level problem (L):

$$\min_{f} L = \sum_{a \in A} \int_0^{f_a} t_a(z, \beta_a) dz, \tag{11}$$

subject to

$$f_a = \sum_{(o,d) \in OD} \sum_{p \in P^{od}} \delta_{ap}^{od} h_p^{od}, \quad \forall a \in A, \tag{12}$$

$$q_{od} = \sum_{p \in P^{od}} h_p^{od} \forall (o,d) \in OD, \tag{13}$$

$$h_p^{od} \geq 0, \quad \forall p \in P^{od}, \quad \forall (o,d) \in OD, \tag{14}$$

$$q_{od} \geq 0, \quad \forall (o,d) \in OD. \tag{15}$$

The solution to the above mathematical problem produces an equilibrium traffic pattern. In the above mathematical equations, constraint (9) requires the capacity improvements at link a be subject to an lower/upper limit $\beta_a^{\min}/\beta_a^{\max}$, while constraint (10) imposes the budgetary constraint.

Equation (12) is the definitional constraint that indicates the relationship between link flow and path flow, while equation (13) indicates the demand conservation constraint.

Equations (14) and (15) indicate that the flow on each link and respectively the $O-D$ demand must be greater or equal to zero.

4. Evolutionary computation in Transportation Network Design problems

Evolutionary computation comprises a set of techniques (genetic algorithms, genetic programming, evolutionary programming and evolutionary strategies) inspired by the evolutionary processes which can be observed in nature: reproduction, mutation, recombination, natural selection and survival of the fittest.

Unlike conventional optimization methods, an evolutionary algorithm operates on a population applying, over the generations, the principles of natural selection and "survival of the fittest" to produce better solution. So it uses in the search process an entire population – possible solutions to the problem – and not just one point in the search space.

The algorithm performs specific operations within a process of reproduction generated by specific operators that are metaphorically linked with their biological correspondents (mutation, crossover, inversion). The qualities of each individual are evaluated by means of special evaluation function (fitness function). The new population (new solutions) selected on the basis of fitness function, which replaces the previous generation, bound for optimal and provides the best solutions for the given issue (Fig. 3).

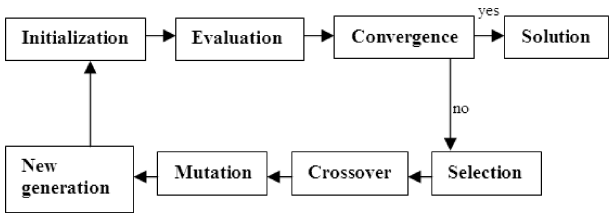

Fig. 3. The basic cycle in a evolutionary algorithm

Every Evolutionary Algorithm needs to find an effective exploration/exploitation ratio of the search space. Exploration is the capability of the algorithm to perform an expansive search within the solution space. Exploitation is the process of using potential solutions already identified to perform more precisely search. Balancing exploration with exploitation in Evolutionary Algorithms is achieved by the selection process, by a proper crossover and mutation operators and by control parameters of the algorithm.

One of the Evolutionary Algorithms which has shown its potential for the solution of many different optimization problems is the Genetic Algorithm (GA). The working method of a GA is very simple and not essentially limited by restrictive assumption (i.e. continuity, derivative conditions, unimodality, etc.). It starts from an initial population: strings (chromosomes) encoding possible solutions of the problem. Each

chromosome corresponds to a possible configuration of the solution. The genetic search is performed from one population to another, using the fitness function values to evaluate the survival capacity of each individual. Each generation, individuals are selected with a probability proportional to their relative fitness, bred using crossover and modified through mutation to form a new population.

These processes finally result in the next generation population of chromosomes that is different from the initial generation. A proper chromosome representation and a good choice of objective function will make the search easier by limiting the search space.

The algorithm ends when the stopping condition (reaching a maximum number of generations or finding an acceptable solution) is met.

Recent studies using GAs have shown their advantages in dealing with Transportation Network Design problems. Hourani et al. [18] addressed a queuing network problem specified in terms of functional, cost, performance and dependability constraints and proposed a solution based on GA. Pattnaik et al. [19] implemented a two phases procedure based on GA for solving Urban Bus Transit Network Design problem, while Karoonsoontawong et al. [20] proposed three metaheuristics for the multiorigin and multidestination CNDP: Simulated Annealing, Genetic Algorithm, and random search. Dung-Ying et al. [21] proposed a solution model based on the use of quantum-inspired GA for dynamic CNDP. Fan et al. [22] formulated the optimal Transit Route Network Design problem with variable transit demand and developed a GA application to solve this TNDP. Other authors [23–25] also developed significant works on this matter, showing that Gas are efficiently in solving Transportation Network Design problems.

4.1. Haploidy vs. diploidy in Genetic Algorithms- relevant research. The most common representation of genetic information in GA contains one string of genetic information for each individual in the population. This is the haploid model that is widely used in evolutionary algorithms [21–25]. In biology, however, genetic material in complex organisms has a diploid or even multiploid chromosome structure. In the diploid structure, each individual has two homologous chromosomes, each containing information for the same functions.

The concept of diploidy in GA is similar with the principles used in human genetics: a diploid genotype comprises a pair of binary strings, called chromosomes. In GA the term "chromosome" refers to a candidate solution to the given optimization problem that can be represented by a set of parameters. These parameters are regarded as the genes of the chromosome, encoded as a bit string.

Holland [26] suggested that the existence of diploidy allows greater genetic diversity in a population, which may increase the population's ability to adapt more quickly to changes in environment over time, compared with haploid coding. He observed that less mutation is needed to maintain a given level of diversity in a diploid GA than a haploid GA.

The effects of the cardinality of genotypic representation have been investigated by GA-community. The results indicate that the diploid method retains greater diversity and shows more robustness than the simple GA. By comparing the frequency distribution of different fitness values in each generation for haploid and diploid individuals, Calabretta et al. [27] concluded that in haploid populations most individuals have an average level fitness and few individuals have a much higher level of fitness. On the other hand, diploid populations have about half of the population with very low level fitness but also tend to include individuals that have average level, good and very good fitness values.

4.2. Implementation and settings. In this study we are improving the complex-encoding GA based on diploid genotype [28] by adding meiosis specific features: duplication and recombination with real valued representation scheme for solution (Fig. 4). Moreover, we are proposing a new mutation operator for imaginary genes of the chromosomes.

```
begin
  t:= 0; initialize population P(0) randomly; evaluate P(0)
  while t ≤Gmax
    //roulette wheel selection
      for all member of population
      r:=random[0,1]; k:=0; partial_sum:=0
      repeat
       k:=k+1;
       partial_sum:=partial_sum + fitness(k);
      until(r< partial_sum / Σ_k fitness(k) or new population is full)
      select_individual:=k
      repeat
    //meiosis
      for all member of population
        mate pairs at random
        //replicate_ chromosome1
              chromatid1:=chromosome1;
              chromatid2:=chromatid1
        //replicate_ chromosome2
              chromatid3:=chromosome2;
              chromatid4 :=chromatid3
        //forming gamete1, gamete2
              // crossover (chromatid1, chromatid3)
        //forming gamete3, gamete4
              //crossover (chromatid2, chromatid4)
    //fertilization
        generate from gametes by randomly selection:
        offspring1
        offspring2
    //non-uniform mutation
        i:=random{1,2,…,N}; randi:=random[0,1]
        if (pmut > randi)
        offspring1[i] ⟶ offspring1mut[i]
        offspring2[i] ⟶ offspring2mut[i]
        endif
      repeat
    evaluate P(t+1)
  repeat
end
```

Fig. 4. The proposed GA for CTND problem

A population of constant size *PS* consisting of diploid chromosomes is given by:

$$\rho_i^{gen} = \left(\rho_{[a_1]i}^{gen} ... \rho_{[a_k]i}^{gen} ... \rho_{[a_m]i}^{gen}\right), \quad gen = 1, \ldots, G_{MAX}, \qquad i = 1, \ldots, PS \tag{16}$$

and

$$\theta_i^{gen} = \left(\theta_{[a_1]i}^{gen} ... \theta_{[a_k]i}^{gen} ... \theta_{[a_m]i}^{gen}\right), \quad gen = 1, \ldots, G_{MAX}, \qquad i = 1, \ldots, PS. \tag{17}$$

For the population size we lean on the values used for other similar network problems [29, 30]. Initially we assume $PS = 20$.

G_{MAX} is the maximum number of generations. Because we use a stopping criterion based on the number of generations, G_{MAX} should be set very large to allow reasonable performance. If the number of generations is too small, there will not be enough chances to find the optimum. Consequently, we set $G_{MAX} = 3000$.

ρ_i^{gen} and θ_i^{gen} indicate the modulus and angle of complex of allele respectively.

Initialization:

The initial population is randomly generated using equations (18) and (19) as follows:

$$\rho_{[a_k]i}^1 = [rand] \times \frac{\beta_a^{\max} - \beta_a^{\min}}{2}, \quad i = 1, \ldots, PS, \tag{18}$$

$$\theta_{[a_k]_i}^1 = [rand] \times 2\pi, \quad i = 1, \ldots, PS. \tag{19}$$

The resultant variable $\beta_{[a_k]_i}^1$ which corresponds to an allele is given by:

$$\beta_{[a_k]_i}^1 = \rho_{[a_k]_i}^1 \times \cos\theta_{[a_k]_i}^1 + \frac{\beta_a^{\min} + \beta_a^{\max}}{2}, \quad i = 1, \ldots, PS. \tag{20}$$

Each resultant chromosome:

$$\beta_i^1 = \left(\beta_{[a_1]i}^1 ... \beta_{[a_k]i}^1 ... \beta_{[a_m]i}^1\right), \quad i = 1, \ldots, PS \tag{21}$$

of the population is a k-dimension vector of capacity improvements: each gene $\beta_{[a_k]_i}^1$ is a real number representing the capacity enhancement associated with link a.

One can note that: $\beta_{[a_k]}^{\min} \leq \beta_{[a_k]_i}^1 \leq \beta_{[a_k]}^{\max}$, $i = 1, \ldots, PS$.

Evaluation:

For each of the leader's decision variables solve the lower problem to obtain the user equilibrium link flows $f_i^{gen} = (...f_{a_i}^{gen}...)$. Then evaluate the program U for each member of the population. A fitter individual implies a lower value for the objective function $U(f_i^{gen}, \beta_i^{gen})$ which is considered as fitness measure of each chromosome β_i^{gen}.

Once the individuals of current population are evaluated according to their fitness, the individuals that will be the parents of the next generation are selected according to the desired selection scheme. This study uses the proportional (roulette wheel) selection.

Next, the selected individuals are paired off randomly to give rise to new offsprings.

The reproduction of the individuals in this study is inspired by the organic mechanism of a meiotic cell division. In this context, the term "meiosis" refers to the process whereby a nucleus divides by two divisions (meiosis I and meiosis II) into four gametes. Meiosis halves the number of chromosomes before sexual reproduction, thereby ensuring that chromosome number does not double with each generation. Before meiosis, each chromosome is replicated, forming two sisters "chromatids" that remain linked together. The two sister chromatids forming each homolog are then separated during the second meiotic division.

In order to perform crossover for two chosen chromatids, both the modules and angles of the arguments are changed as described below.

Let us consider that we perform the crossover operator for chromatid 1 and chromatid 3.

Their modules are:

$$\rho_1 = (\rho_{[a_1]1} ... \rho_{[a_k]1} ... \rho_{[a_m]1})$$

and respectively

$$\rho_3 = (\rho_{[a_1]3} ... \rho_{[a_k]3} ... \rho_{[a_m]3})$$

and their angles are:

$$\theta_1 = (\theta_{[a_1]1} ... \theta_{[a_k]1} ... \theta_{[a_m]1})$$

and respectively

$$\theta_3 = (\theta_{[a_1]3} ... \theta_{[a_k]3} ... \theta_{[a_m]3}).$$

After performing their crossover, two gametes are obtained as following:

$$\text{gamete}_1 \text{ has } \begin{cases} \rho_{c_1} = (...., r \times \rho_{[a_k]1} + (1-r) \times \rho_{[a_k]3}, ...); \\ \theta_{c_1} = (...., r \times \theta_{[a_k]1} + (1-r) \times \theta_{[a_k]3}, ...) \end{cases} \tag{22}$$

$$\text{gamete}_2 \text{ has } \begin{cases} \rho_{c_2} = (...., (1-r) \times \rho_{[a_k]1} + r \times \rho_{[a_k]3}, ...); \\ \theta_{c_2} = (...., (1-r) \times \theta_{[a_k]1} + r \times \theta_{[a_k]3}, ...), \end{cases} \tag{23}$$

where r is a random number between 0 and 1.

The probability of crossover is p_c, so that an average of $p_c \times 100\%$ chromosomes undergo crossover.

Fertilization (putting together two gametes resulted from meiosis) is done by randomly combining gametes from the gene pool: for each parent individual, two of the gametes from the four that have been formed are then selected randomly to form two new offsprings.

The next genetic operator, mutation, is a mechanism for extending the search on the new areas of search space. Mutation modifies the genotype, and thus the phenotype, by random altering of bit's values inside chromosome with given probability.

In this paper we use non-uniform mutation [31] for the module of an argument. Let us consider that we perform the non-uniform mutation operator for a resultant offspring.

If $\rho_{=}(\rho_{[a_1]} ... \rho_{[a_k]} ... \rho_{[a_m]})$ is the modulus component and $\rho_{[a_k]}$ is selected at random for mutation, the result is:

$$\rho^{mut} = (\rho_{[a_1]} ... \rho_{[a_k]}^{mut} ... \rho_{[a_m]}),$$

where

$$\rho_{[a_k]}^{mut} = \begin{cases} \rho_{[a_k]} + (\beta_{a_k}^{\max} - \rho_{[a_k]}) f(gen) \text{ if } r_1 < 0.5, \\ \rho_{[a_k]} + (\rho_{[a_k]} - \beta_{a_k}^{\min}) f(gen) \text{ if } r_1 > 0.5, \\ \rho_{[a_k]}, \text{ otherwise,} \end{cases} \tag{24}$$

where

$$f(gen) = \left(r_2 \left(1 - \frac{gen}{G_{\max}} \right) \right)^{\tau}, \tag{25}$$

r_1, r_2 are randomly generated numbers in interval (0, 1); *gen* is the current generation; τ is a system parameter determining the degree of dependency on the iteration number. In this study we set $\tau = 3$.

For an angle of an argument we propose a new mutation operator defined as follows: if $\theta = (\theta_{[a_1]} ... \theta_{[a_k]} ... \theta_{[a_m]})$ is the angle component and $\theta_{[a_k]}$ is selected at random for mutation, the result is: $\theta^{mut} = (\theta_{[a_1]} ... \theta_{[a_k]}^{mut} ... \theta_{[a_m]})$, where:

$$\theta_{[a_k]}^{mut} = \begin{cases} \max(\theta_{[a_k]} - f(gen) \cdot \pi, _0) \text{ if } r_1 < 0.5, \\ \min(\theta_{[a_k]} + f(gen) \cdot \pi, 2\pi) \text{ if } r_1 > 0.5, \\ \theta_{[a_k]}, \text{ otherwise,} \end{cases} \tag{26}$$

where

$$f(gen) = 1 - r_2^{\left(1 - \frac{gen}{G_{\max}}\right)^{\tau}} \tag{27}$$

and the other parameters are the same as those described above.

As can be seen from Eqs. (25) and (27), the amplitude of the change decrease as one approaches the maximum number of generations. Thus, these mutation operators perform global search during the initial search and local search in the later generations. Moreover, the local searching ability of the algorithm is improved, as well as the algorithm's efficiency.

A large number of studies which explore the interaction among different GA parameters showed that in general GAs will work well with high crossover & low mutation probability. Therefore, common to each run were the following parameter settings: population size was 20, crossover rate was 80%, and mutation rate was 3%.

4.3. Application study. In order to verify the effectiveness of the model, the proposed solution method is implemented into a test network. The numerical experiment involves a road network taken from [17] as shown in Fig. 5. This network contains 6 nodes and two $O-D$ pairs, 1→6 and 6→1. The network information is summarized in Table 1 and the complete data relating to this example can be found in [17].

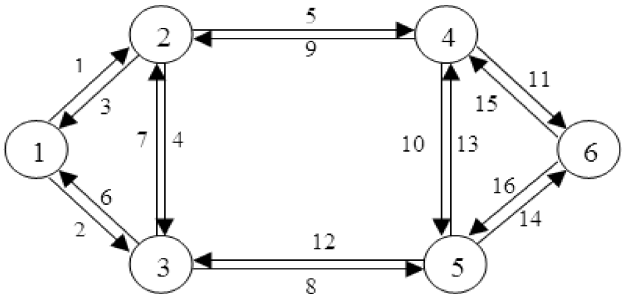

Fig. 5. Test Network

Table 1
Network parameter for the test network

Arc (a)	t_a^0	b_a	w_a	d_a
(1,2)	1.0	10.0	3.0	2.0
(1,3)	2.0	5.0	10.0	3.0
(2,1)	3.0	3.0	9.0	5.0
(2,3)	4.0	20.0	4.0	4.0
(2,4)	5.0	50.0	3.0	9.0
(3,1)	2.0	20.0	2.0	1.0
(3,2)	1.0	10.0	1.0	4.0
(3,5)	1.0	1.0	10.0	3.0
(4,2)	2.0	8.0	45.0	2.0
(4,5)	3.0	3.0	3.0	5.0
(4,6)	9.0	2.0	2.0	6.0
(5,3)	4.0	10.0	6.0	8.0
(5,4)	4.0	25.0	44.0	5.0
(5,6)	2.0	33.0	20.0	3.0
(6,4)	5.0	5.0	1.0	6.0
(6,5)	6.0	1.0	4.5	1.0

Demand is equal to D from node 1 to node 6 and 2D from node 6 to node 1. It is multiplied by scalars to represent increasing demand.

Table 2 summarizes two travel demand scenarios for this network in order to observe the effect of traffic congestion. A matrix E representing link interaction was randomly generated to obtain suggestive input data and is given in Table 3.

Table 2
The levels of traffic demand

Case	Traffic demand D from node 1 to node 6	Traffic demand D from node 6 to node 1	Total demand	Upper limit of capacity improvements
I	5.0	10.0	15.0	$0 \leq \beta_a \leq 10, \forall a \in A$
II	10.0	20.0	30.0	$0 \leq \beta_a \leq 20, \forall a \in A$

Table 3
Link interactions

$\varepsilon_{a,b}$	(1,2)	(1,3)	(2,1)	(2,3)	(2,4)	(3,1)	(3,2)	(3,5)	(4,2)	(4,5)	(4,6)	(5,3)	(5,4)	(5,6)	(6,4)	(6,5)
(1,2)	1.00	0.020	0.011	0.13	0.06	0.12	0.03	0.00	0.01	0.10	0.00	0.00	0.01	0.02	0.14	0.09
(1,3)	0.10	1.00	0.12	0.18	0.00	0.08	0.09	0.07	0.00	0.02	0.00	0.20	0.00	0.00	0.03	0.00
(2.1)	0.04	0.13	1.00	0.11	0.03	0.18	0.17	0.12	0.15	0.02	0.00	0.06	0.00	0.01	0.03	0.19
(2.3)	0.00	0.01	0.19	1.00	0.10	0.20	0.03	0.00	0.03	0.00	0.00	0.10	0.07	0.15	0.01	0.00
(2.4)	0.15	0.00	0.18	0.17	1.00	0.00	0.12	0.01	0.19	0.01	0.04	0.00	0.17	0.14	0.07	0.00
(3.1)	0.17	0.12	0.15	0.19	0.00	1.00	0.18	0.14	0.00	0.00	0.10	0.16	0.00	0.00	0.00	0.05
(3.2)	0.14	0.19	0.17	0.06	0.11	0.12	1.00	0.09	0.19	0.00	0.00	0.19	0.00	0.04	0.01	0.03
(3.5)	0.06	0.18	0.07	0.19	0.02	0.15	0.19	1.00	0.03	0.12	0.00	0.17	0.12	0.08	0.00	0.14
(4.2)	0.16	0.01	0.16	0.12	0.14	0.00	0.12	0.00	1.00	0.18	0.15	0.01	0.14	0.00	0.17	0.00
(4.5)	0.01	0.00	0.02	0.05	0.14	0.01	0.09	0.19	0.12	1.00	0.14	0.02	0.17	0.16	0.13	0.18
(4.6)	0.07	0.03	0.02	0.09	0.12	0.00	0.00	0.07	0.19	0.11	1.00	0.00	0.12	0.15	0.19	0.17
(5.3)	0.00	0.18	0.00	0.09	0.00	0.18	0.19	0.16	0.02	0.16	0.19	1.00	0.11	0.12	0.00	0.16
(5.4)	0.01	0.00	0.09	0.08	0.16	0.01	0.00	0.19	0.14	0.14	0.11	0.18	1.00	0.14	0.18	0.12
(5.6)	0.00	0.00	0.08	0.09	0.03	0.07	0.08	0.12	0.00	0.16	0.13	0.19	0.15	1.00	0.14	0.16
(6.4)	0.00	0.09	0.00	0.08	0.19	0.00	0.00	0.07	0.14	0.15	0.19	0.00	0.12	0.17	1.00	0.13
(6.5)	0.00	0.02	0.00	0.00	0.03	0.10	0.01	0.14	0.02	0.17	0.12	0.14	0.09	0.08	0.18	1.00

4.4. Result analysis. Results are presented comparing the performance of the proposed algorithm and the traditional/simple Genetic Algorithm.

Another test was performed for the proposed CTND problem using the Statistical Genetic Algorithm [32] with binary representation.

In this test, for $\beta_{[a_i]} \in [\beta^{\min}_{[a_i]}, \beta^{\max}_{[a_i]}]$ represented by $\{a_1, \ldots, a_N\} \in \{0,1\}^N$, $i = 1, .., m$, $T : \{0,1\}^N \to [\beta^{\min}_{[a_i]}, \beta^{\max}_{[a_i]}]$ defines the representation:

$$T(a_1, \ldots, a_N) = \beta^{\min}_{[a_i]} + \\ + \frac{\beta^{\max}_{[a_i]} - \beta^{\min}_{[a_i]}}{2^N - 1} \left(\sum_{j=0}^{N-1} \beta_{[a_{N-j}]} 2^j \right) \in [\beta^{\min}_{[a_i]}, \beta^{\max}_{[a_i]}]. \quad (28)$$

The sensitivity analysis was performed on Statistical GA parameters to determine their influence on the algorithm's performance. After experimenting with various parameter combinations, we decided to adopt the following parameter values in both simple GA and Statistical GA:

– size of population PS: 100;
– crossover rate $p_c = 0.8$;
– total number of generations $G_{MAX} = 2000$.

The results of this comparison are summarized in Figs. 6–7 and in Tables 4–6. We use the same random number generator seed for all the GA runs. That generates the same starting population and provides a fairer comparison of the proposed implementations. Also for comparison purpose all algorithms are programmed in Matlab 6.0 version environment.

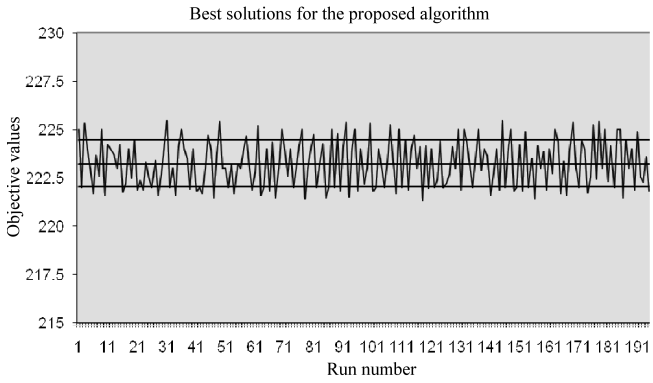

Fig. 6. Best objective values for the proposed algorithm – 100 runs, case I

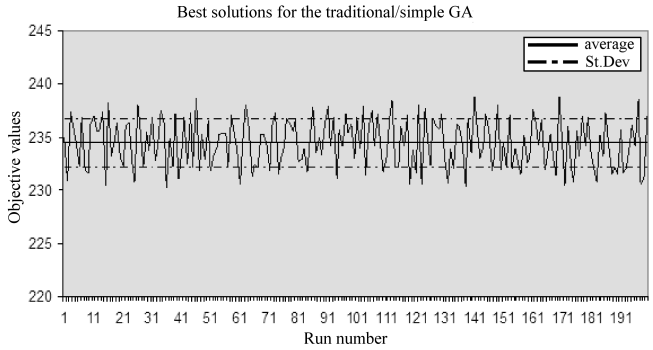

Fig. 7. Best objective values for the traditional/simple GA– 100 runs, case I

Details of the resultant link capacity enhancements and corresponding objective function values obtained by these methods are shown in Tables 4 and 5.

Table 4
Arc improvement solutions and objective value for case I

Capacity enhancement	Simple GA	Statistical GA	Proposed algorithm
β_1|Arc (1.2)			
β_2| Arc (1.3)		1.084	0.142
β_3| Arc (2.1)	1.245	0.173	0.579
β_4| Arc (2.3)			
β_5| Arc (2.4)			
β_6| Arc (3.1)	5.89	5.499	6.238
β_7| Arc (3.2)	0.243	0.384	
β_8| Arc (3.5)			
β_9| Arc (4.2)			
β_{10}| Arc(4.5)		0.114	
β_{10}| Arc(4.6)			
β_{12}| Arc(5.3)		0.039	0.058
β_{13}| Arc(5.4)			
β_{14}| Arc(5.6)			
β_{15}| Arc(6.4)	0.249		
β_{16}| Arc(6.5)	6.784	6.704	6.112
Objective value	**230.210**	**224.570**	**221.340**

Table 5
Arc improvement solutions and objective value for case II

Capacity enhancement	Simple GA	Statistical GA	Proposed algorithm
β_1|Arc (1.2)			
β_2| Arc (1.3)	5.392	5.117	4.851
β_3| Arc (2.1)	10.114	11.014	9.304
β_4| Arc (2.3)			
β_5| Arc (2.4)			
β_6| Arc (3.1)	7.129	9.142	10.948
β_7| Arc (3.2)	0.763	0.544	
β_8| Arc (3.5)	0.214	0.179	0.815
β_9| Arc (4.2)	0.175	0.214	0.025
β_{10}| Arc(4.5)		0.103	0.039
β_{10}| Arc(4.6)			
β_{12}| Arc(5.3)			
β_{13}| Arc(5.4)			
β_{14}| Arc(5.6)			1.107
β_{15}| Arc(6.4)	9.814	3.489	4.079
β_{16}| Arc(6.5)	17.495	18.944	16.438
Objective value	**611.443**	**593.708**	**587.124**

Table 6
Performances of the proposed algorithm and the traditional/simple GA for case I

Traditional/simple GA				Statistical GA				Proposed GA			
100 runs	Hits	Average	St. Dev	100 runs	Hits	Average	St. Dev	100 runs	Hits	Average	St. Dev
[230.21.238.79]	7	234.42	2.28	[224.56.230.16]	15	227.09	1.36	[221.34.225.47]	16	223.24	1.22

Table 6 includes the average, maximum, minimum and standard deviation obtained for case I after performing 100 independent runs. "Hits" is the number of runs in which we obtained a solution differing by less than 0.1% from the best solution obtained. "Average" means the best objective function value of the 100 runs of each algorithm. It shows the quality of candidate solutions through iterations. The difference between max and min objective values expresses the search range of the algorithms. "St. Dev" denotes standard deviation, which expresses the searching capacity of each algorithm. As table 6, the proposed algorithm has a smaller mean value and a smaller standard deviation. That is, this algorithm has better convergence stability, is more robust and is able to reach good solutions across 100 runs.

Figure 6 shows the best objective values obtained by the 100 runs performed when using the proposed algorithm for case I, while Fig. 7 shows the best objective values obtained by the 100 runs performed when using the traditional/simple GA for the same case. The average and standard deviation for these 100 runs are shown in the figures. It is not difficult to observe that the proposed methodology gives better results and generates solutions with significantly lower cost.

The population size and the number of generations have a large influence on the GA's performance. In order to determine the appropriate population size for the proposed methodology, different population sizes of 20, 30 and 50 individuals are tested, respectively for case I. Figure 8 shows that although the convergence rate increases when the population size increases, the improvement is not obvious. Consequently, the population size of 20 individuals will be suitable.

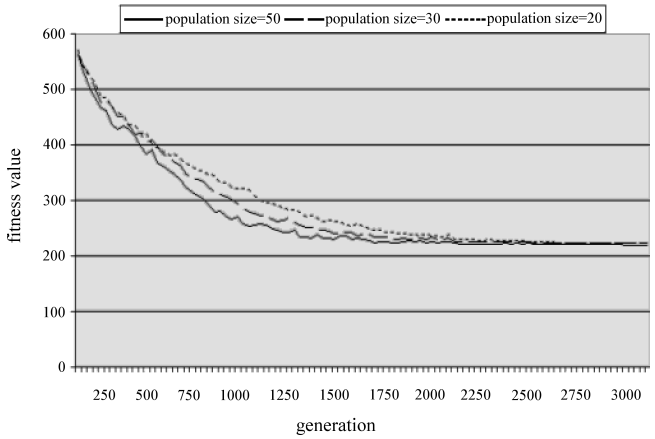

Fig. 8. Average fitness of populations by generation, case I

From the results one can observe an intense decrease of fitness scores in early generations, then the decrease fluctuates and then fitness scores become stable in later generations. The algorithm finds an optimal solution within less than 3000 generations that leads to less computational time for the solution process. This analyze of the influence of the changing parameter (population size) may have further consequences on designing other evolutionary algorithms for this problem.

5. Conclusions

In this study we have extended the CTNDP frame to incorporate an important feature for particular applications: interactions among different links. This broaden model is formulated as a bi-level mathematical model. Since it is very difficult to obtain an exact optimal solution for such a problem, a heuristic search procedure is used to find near optimal solutions. The essential features of our proposed algorithm include a diploid based complex-encoding with meiosis specific features and a novel mutation operator that performs global search during the initial search and local search in the later generations.

The implementation on a test network verifies the effectiveness of the proposed method.

This approach merits further study concerning the analysis for other existing selection, crossover and mutation operators. Another possible extension would be for solving network design problems from a broader perspective of dynamic traffic, elastic demand and other uncertainties related to supply, route capacities and processing/transportation costs.

REFERENCES

[1] J.G. Wardrop, "Some theoretical aspects of road traffic research", *Proc. Institute of Civil Engineers* II, 325–378 (1952).

[2] C.F. Daganzo and Y. Sheffi, "On stochastic models of traffic assignment", *Transportation Science* 11 (3), 253–274 (1977).

[3] L.N. Vicente and P.H Calamai, "Bilevel and multilevel programming. A bibliography review", *J. Global Optimization* 5, 291–306 (1994).

[4] M. Chen and A.S. Alfa, "Algorithms for solving Fisk's stochastic traffic assignment model", *Transportation Science* 25 (6), 405–411 (1991).

[5] Z. Drezner and G.O. Wesolowsky, "Selecting an optimum configuration of one-way and two-way routes", *Transportation Science* 31, 386–394 (1997).

[6] S.N. Kuan, H.L.Ong, and K.M. Ng, "Applying metaheuristics to feeder bus network design problem", *Asia-Pacific J.Operational Research* 21 (4), 543–560 (2004).

[7] M.Y. Abdulaal and L.J. LeBlanc, "Continuous equilibrium network design models", *Transportation Research* B 13, 19–32 (1979).

[8] T.L. Friesz, H. Cho, N.Y. Mehta, and R.Tobin, "Simulated annealing methods for network design problems with variational inequality constraints", *Transportation Science* 26 (1), 18–26 (1992).

[9] Z.Y. Gao, H. Sun, and L. Shan, "A continuous equilibrium network design model and algorithm for transit systems", *Transportation Research* B 38, 235–250 (2004).

[10] D.A. Hensher, K.J. Button, K.E. Haynes, and P.R. Stopher, *Handbook of Transport Geography and Spatial System*, Elsevier, Amsterdam, 2004.

[11] A. Nagurney, *Spatial Equilibration in Transport Networks: Handbook of Transport Geography and Spatial Systems*, Elsevier, Amsterdam, 2003.

[12] Bureau of Public Roads, *Traffic Assignment Manual*, Dept. of Commerce, Urban Planning Division, Washington, 1964.

[13] Federal Highway Administration, *Traffing Monitoring Guide*, http://www.fhwa.dot.gov/ohim/tmguide/index.htm (2011).

[14] C. Hurkens, J. Keijsper, and L. Stougie, "Virtual private network design: a proof of the tree routing conjecture on ring networks", *Siam J. Discrete Mathematics* 21, 482–503 (2007).

[15] K. Peric and M. Boile, "A combined model for intermodal networks with variable transit frequencies", *Proc. 85th Annual Transportation Research Board Meeting, Washington* SAD 1, 136–145 (2006).

[16] P. Marcotte and G. Marquis, "Efficient implementation of heuristics for the continuous network design problem", *Annals of Operations Research* 34, 163–176 (1992).

[17] C. Suwansirikul, T.L. Friesz, and R.L. Tobin, "Equilibrium decomposed optimization: a heuristic for the continuous equilibrium network design problem", *Transportation Science* 21, 254–263 (1987).

[18] H. Mourani, M. Ozden, P. Maynard-Zhang, and F. Moore, "Applying genetic algorithms to queuing network design", *J. Automation and Systems Engineering* 1 (1), 32–39 (2007).

[19] S.B. Pattnaik, S. Mohan, and V.M. Tom, "Urban bus transit network design using genetic algorithm", *J. Transportation Engineering* 124 (4), 368–375 (1998).

[20] A. Karoonsoontawong and S. T. Waller, "Dynamic continuous network design problem: linear bi-level programming and metaheuristic approaches", *J. Transportation Research Board* 1964, 104–117 (2006).

[21] L. Dung-Ying and S. T. Waller, "A quantum-inspired genetic algorithm for dynamic continuous network design problem", *Int. J. Transportation Research* 1 (1), 81–93 (2009).

[22] W. Fan and R. Machemehl, "Optimal transit route network design problem with variable transit demand: genetic algorithm approach", *J. Transportation Engineering* 132, 40–51 (2006).

[23] S. Ngamchai and D.J. Lovell, "Optimal time transfer in bus transit route network design using a genetic algorithm", *J. Transportation Engineering* 129 (5), 510–521 (2003).

[24] K. Jeon, J.S. Lee, S. Ukkusuri, and S.T. Waller, "New approach for relaxing computational complexity of discrete network design problem using selectorecombinative genetic algorithm," *Proc. Transportation Research Board's 85th Annual Meeting* 1, CD-ROM (2006).

[25] Y.E. Xiong and J.B. Schneider, "Transportation network design using a cumulative genetic algorithm and neural network", *Transportation Research Record* 1364, 37–44 (1993).

[26] J.H. Holland, "Adaptation in natural and artificial systems: an introductory analysis with applications to biology", *Control and Artificial Intelligence* 1, CD-ROM (1992).

[27] R. Calabretta, R. Galbiati, S. Nolfi, and D. Parisi, "Two is better than one: a diploid genotype for neural networks", *Neural Processing Letters* 4, 1–7 (1996).

[28] Y. Wang, S. Gao, H. Zhang, and Z. Tang, "An improved genetic algorithm based complex-valued encoding", *IJCSNS Int. J. Computer Science and Network Security* 10 (6), 168–174 (2010).

[29] F.A. Kidwai, B.R. Marwah, K. Deb, and M.R. Karim, "A genetic algorithm based bus scheduling model for transit network", *Proc. Eastern Asia Society for Transportation Studies* 5, 447–489 (2005).

[30] S. Dinu and R. Pacuraru, "An intelligent modeling method based on genetic algorithm for partner selection in virtual organizations", *Business and Economic Horizons* 5, CD-ROM (2011).

[31] Z. Michalewicz, T. Logan, and S. Swaminathan, "Evolutionary operators for continuous convex parameter space", *Proc. Third Annual Conf. on Evolutionary Programming* 1, 84–97, (1994).

[32] M.A. Tabarzad, C. Lucas, and A. Hamzeh, "Statistical genetic algorithm", *Trans. on Engineering, Computing and Technology* 14, 100–104 (2006).