Artificial bee colony algorithm for continuous network design problem with link capacity expansions

O. Baskan[1], M. Dell'Orco[2]

[1]Department of Civil Engineering, Pamukkale University, Denizli, Turkey, obaskan@pau.edu.tr
[2]Department of Water Engineering and Chemistry, Technical University of Bari, Italy, dellorco@poliba.it

Abstract

In general, Continuous Network Design Problem (CNDP) is concerned with the optimal capacity expansion of existing links in a given network. The measure of network performance can be described as the sum of total travel times and the investment cost, converted to travel time, for capacity expansions of links. Due to the non-convex characteristics of CNDP, the problem is formulated as a bi-level programming problem. Although proposed algorithms in literature are capable of solving CNDP for a given road network, an efficient algorithm, which is capable of finding the global or near global optima of the upper level decision variables of CNDP, is still needed. Therefore, this paper deals with the finding of optimal link capacity expansions in a given road network using the Artificial Bee Colony (ABC) algorithm. More precisely, a bi-level method has been proposed, in which the lower level problem is formulated as a user equilibrium traffic assignment model, solved by the ABC. To show the effectiveness of the ABC algorithm, the example road network is used for different size of travel demand. Results produced by the ABC algorithm are compared with the results obtained by Simulated Annealing (SA) and Genetic Algorithm (GA) taken from literature. According to the results, the ABC algorithm can be used effectively to solve CNDP with link capacity expansions within much less number of traffic assignments than SA and GA.

Keywords: *Continuous network design problem, Artificial bee colony algorithm, Bi-level programming*

1 Introduction

The Continuous Network Design Problem (CNDP) is at the core of transportation problems and has been extensively studied in the literature aiming to determine the optimal capacity expansions for a set of selected links in a given transportation network. The measure of network performance can be described as the sum of total travel times and the investment cost of link capacity expansions, converted to travel time. Determining the global or near global optimum solution is of great importance in CNDP. Due to the non-convex and non-smooth characteristics of the CNDP, it is formulated as bi-level programming problem. In CNDP, upper level can be modelled with an objective function defined as the sum of total travel time and total investment cost of link capacity expansions in a given network, whilst the lower level is formulated as a traffic assignment model.

Abdulaal and LeBlanc (1979) were the first ones to solve CNDP problem for medium-sized realistic network using Hooke-Jeeves (HJ) algorithm. Suwansirikul et al. (1987) proposed the Equilibrium Decomposition Optimization (EDO) for finding an approximate solution to CNDP, and tested this heuristic on several networks. Marcotte and Marquis (1992) presented efficient implementations of heuristic procedures in small-sized networks for solving CNDP. Furthermore, Friesz et al. (1992) used Simulated Annealing (SA) approach to solve CNDP, and found that the proposed heuristic is more efficient than the Iterative Optimization Assignment (IOA) algorithm, HJ algorithm, and EDO approach for finding the global optimum solution. Friesz et al. (1993) presented a model for continuous multiobjective optimal design of a transportation network. The results showed that SA is ideally suited for solving multi objective versions of the equilibrium network design problem. Davis (1994) used the generalized reduced gradient method and sequential quadratic programming to solve CNDP.

Meng et al. (2001) converted the bi-level program of CNDP to a single level continuously differentiable optimization problem. Chiou (2005) used a bi-level programming technique to formulate CNDP. Four variants of gradient-based methods are presented and numerical comparisons are made with several test networks to solve CNDP. Similarly, Ban et al. (2006) proposed a relaxation method to solve CNDP when the lower level is a nonlinear complementary problem and obtained good results. Xu et al. (2009) used SA and Genetic Algorithm (GA) methods to find optimal solutions of CNDP. They found that when demand is large, SA is more efficient than GA in solving CNDP, and much more computational time is needed for GA to achieve the same optimal solution as SA.

Although the proposed algorithms existed in literature are capable of solving CNDP for a given road network, an efficient algorithm, which is capable of finding the global or near global optimum solutions of the upper level optimization problem of CNDP, is still needed. It is also important to compare the efficiency of the available heuristic methods in literature, to find the most efficient ones. In fact, to evaluate the upper level objective function, a traffic assignment problem has to be solved in lower level and, to do this for a real road network, the computation time is considerably large. Therefore, this paper deals with the finding optimal link capacity expansions in a given road network using Artificial Bee Colony (ABC) algorithm. For this purpose, a bi-level model has been proposed, in which the lower level problem is formulated as User Equilibrium (UE) traffic assignment model and Frank-Wolfe (FW) method is used to solve it.

The rest of this paper is organized as follows. The notations are given in Section 2. Problem formulation for CNDP subject to link capacity expansions is given in Section 3. In the next section, ABC algorithm is presented for finding the upper level decision variables for CNDP. In Section 5, numerical calculations are conducted on example road network. Concluding remarks and future study directions are given in Section 6.

2 Notations

$\boldsymbol{A}$	the set of links, $\forall a \in \boldsymbol{A}$
$\boldsymbol{c_{rs}}$	the set of minimum path travel times between O-D pair rs $\forall r \in \boldsymbol{R},\ s \in \boldsymbol{S}$
$\boldsymbol{D}$	the vector of O-D pair demands, $\boldsymbol{D} = [\boldsymbol{D_{rs}}]\ \forall r \in \boldsymbol{R},\ s \in \boldsymbol{S}$
$\boldsymbol{f}$	the vector of path flows, $\boldsymbol{f} = [f_k^{rs}], \forall r \in \boldsymbol{R},\ s \in \boldsymbol{S},\ k \in \boldsymbol{K_{rs}}$
$\boldsymbol{g}$	the vector of investment costs, $\boldsymbol{g} = [g_a(y_a)]\ \forall a \in \boldsymbol{A}$
$\boldsymbol{K_{rs}}$	the set of paths between O-D pair $rs \forall r \in \boldsymbol{R},\ s \in \boldsymbol{S}$
$\boldsymbol{R}$	the set of origins
$\boldsymbol{S}$	the set of destinations
$\boldsymbol{t}$	the vector of link travel times, $\boldsymbol{t} = [t_a(x_a, y_a)]\ \forall a \in \boldsymbol{A}$
$\boldsymbol{u}$	the vector of upper bound for link capacity expansions, $\boldsymbol{u} = [u_a], \forall a \in \boldsymbol{A}$
$\boldsymbol{x}$	the vector of equilibrium link flows, $\boldsymbol{x} = [x_a], \forall a \in \boldsymbol{A}$
$\boldsymbol{y}$	the vector of link capacity expansions, $\boldsymbol{y} = [y_a], \forall a \in \boldsymbol{A}$
Z	upper level objective function
z	lower level objective function
ρ	the conversion factor from investment cost to travel times
$\delta_{a,k}^{rs}$	the link/path incidence matrix variables, $\forall r \in \boldsymbol{R},\ s \in \boldsymbol{S},\ k \in \boldsymbol{K_{rs}}, a \in \boldsymbol{A}$. $\delta_{a,k}^{rs} = 1$ if route k between O-D pair rs uses link a, and $\delta_{a,k}^{rs} = 0$ otherwise
τ	the set of auxiliary link flows
α_a, β_a	the parameters of link cost function, $\forall a \in \boldsymbol{A}$
θ_a	the link capacity, $\forall a \in \boldsymbol{A}$
ϕ_n	the step size at n*th* iteration

3 Problem Formulation

The CNDP can be represented within the framework of a leader-follower or Stackelberg game, where the supplier is the leader and the user is the follower (Fisk, 1984). It is assumed that the leader as transportation planning manager can influence the user's path choice behavior but cannot control it. The users make their decision in a user optimal manner under the given service level of transportation networks (Gao et al., 2007). The network planners of the upper level are assumed to make the decisions about the improvement of link capacities and investments to minimize the total system cost. This interaction between two groups can be formulated as bi-level programming model. Therefore, the bi-level programming model is used to solve CNDP with link capacity expansions in this study. The upper level objective function for CNDP can be formulated as follows:

$$\min_{y} Z(x,y) = \sum_{a\in \boldsymbol{A}} (t_a(x_a(y), y_a)x_a(y) + \rho g_a(y_a)) \tag{1}$$

$$\text{s.t.} \qquad 0 \le y_a \le u_a\ , \quad \forall a \in \boldsymbol{A}$$

The constraint ensures that the investment cost of link $a \in \boldsymbol{A}$ will not exceed the related budget. It is also the non-negativity constraint of the decision variables.

In general, the CNDP models assume that the demand is given and fixed, and the user's route choice is characterized by the UE assignment. The UE assignment problem at the lower level can be represented as follows:

$$\min_{x} z = \sum_{a\in \boldsymbol{A}} \int_{\boldsymbol{0}}^{\boldsymbol{x_a}} t_a(w, y_a)dw \tag{2}$$

$$\begin{aligned}
\text{s.t.} \quad & \sum_{k\in \boldsymbol{K}} f_k^{rs} = D_{rs} && \forall r \in \boldsymbol{R},\ s \in \boldsymbol{S},\ k \in \boldsymbol{K}_{rs} \\
& x_a = \sum_{rs} \sum_{k\in K_{rs}} f_k^{rs}\delta_{a,k}^{rs} && \forall r \in \boldsymbol{R},\ s \in \boldsymbol{S},\ a \in \boldsymbol{A}, k \in \boldsymbol{K}_{rs} \\
& f_k^{rs} \ge 0 && \forall r \in \boldsymbol{R},\ s \in \boldsymbol{S},\ k \in \boldsymbol{K}_{rs}
\end{aligned}$$

where the constraints are definitional, conservation of the flow constraints and non-negativity, respectively. For solving UE assignment, the still most widely used solution algorithm is the FW algorithm. It was actually proposed for solving quadratic optimization problems with linear constraints (Frank and Wolfe, 1956). This method is especially useful for determining the equilibrium flows for transportation networks, since the direction finding step can be executed in a relatively efficient way. Therefore, we have used this algorithm for finding user equilibrium link flows in lower level optimization problem. It can be summarized as follows:

Step 0: Set n=1. Get x_a^n through the all-or-nothing assignment, based on $t_a = t_a(0),\ \forall a \in \boldsymbol{A}$.

Step 1: Set $t_a^n = t_a(x_a^n),\ \forall a \in \boldsymbol{A}$

Step 2: Perform all-or-nothing assignment based on t_a^n. This provides a set of auxiliary flows τ_a^n.

Step 3: Use golden section algorithm (see for details Sheffi, 1985), to find ϕ_n that solves:

$$\min_{0\le\phi\le 1} \sum_{a} \int_{0}^{x_a^n + \phi(\tau_a^n - x_a^n)} t_a(w)dw \tag{3}$$

Step 4: Set $x_a^{n+1} = x_a^n + \phi_n(\tau_a^n - x_a^n),\ \forall a \in \boldsymbol{A}$

Step 5: Convergence test. If the stopping criterium given below in Eq. (4) is met, terminate the algorithm; otherwise set $n = n+1$ and go to Step 1.

$$\sum_{rs} \frac{\left| c_{rs}^{n} - c_{rs}^{n-1} \right|}{c_{rs}^{n}} \leq 0.10 \tag{4}$$

where c_{rs}^{n} is the minimum path travel time between O-D pair *rs* at the n*th* iteration.

4 Artificial Bee Colony (ABC) Algorithm

The foraging behaviors of honeybees have recently been one of the most interesting research areas in swarm intelligence. Some approaches have been proposed to model the specific intelligent behaviours of honey bee swarms and they have been applied for solving optimization problems. Tereshko (2000) considered a bee colony as a dynamical system gathering information from an environment and adjusting its behaviour in accordance to it. Lucic and Teodorovic (2002) and Teodorovic (2003) suggested to use bee swarm intelligence aimed at solving complex problems in traffic and transportation. Teodorovic and Dell'Orco (2005) proposed the bee colony optimization to solve combinatorial problems characterized by uncertainty, as well as deterministic combinatorial problems. Teodorovic and Dell'Orco (2008) presented an application of the bee colony optimization, efficient in solving the ride-matching problem. Their results showed that proposed metaheuristic appears very promising, and indicated that the development of new models based on swarm intelligence principles could significantly contribute to the solution of a wide range of complex engineering and management problems. The Artificial Bee Colony (ABC) algorithm is a new population-based metaheuristic approach proposed by Karaboga (2005) and further developed by Karaboga and Basturk (2007a, 2007b, 2008). It is inspired by the foraging behavior of honeybee swarm. The foraging bees are classified into three categories employed, onlookers and scouts. Half of the bee colony consists of employed bees, and another half consists of onlookers. In ABC algorithm, the position of a food source represents a possible solution to the optimization problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution (Karaboğa and Akay, 2009).The number of employed bees or the onlooker bees is equal to the number of solutions in the population. Employed bees are responsible for searching available food sources and gathering required information. They also pass their food information to onlooker bees. The onlookers select good food sources from those found by the employed bees to further search the foods. When the quality of the food source is not improved through a predetermined number of cycles, the food source is abandoned by its employed bee. In this case, the employed bee becomes a scout and starts to search for a new food source in the vicinity of the hive.

In ABC algorithm, each cycle of the search consists of three steps. At the initialization step, the ABC algorithm generates a randomly distributed initial population as number of *SN*, where *SN* denotes the number of employed bees or onlooker bees. Each initial solution $x_i\ (i = 1,2,.....,SN)$ is a *D*-dimensional vector which *D* is the number of decision variables of a given optimization problem. At the first step of the cycle, employed bees come into the hive and share with the bees waiting on the dance area information about nectar sources. A bee waiting on the dance area is called onlooker, and is responsible for making decision about the choice of a food source. At the second step, an onlooker chooses a food source area depending on the nectar information distributed by the employed bees on the dance area. As the nectar amount of a food source increases, the probability of choice of that food source increases as well. The determination of a new food source is carried out by the bees based on a visual comparison process of positions of food sources. At the third step of the cycle, when a food source is abandoned by the bees, a new food source is randomly determined by a scout bee and replaces the abandoned one. These three steps are repeated until a predetermined number of cycles, called Maximum Cycle Number (MCN), is reached. An onlooker bee chooses a food source depending on the probability value, p_i, as follows:

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{5}$$

where fit_i is the fitness value of solution i. In this way, the employed bees exchange their information with the onlookers. In order to share the information of nectar amount of the food sources, the employed bees use a proportional selection method known as "roulette wheel selection".

In order to produce a candidate food location from the old one in population, the following equation is used.

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{6}$$

where v_{ij} is the candidate food position which can replace the old one in the memory and ϕ_{ij} is a random number generated in the interval [-1,1]. The values $k = 1,2,\ldots,SN$ and $j = 1,2,\ldots,D$ are randomly chosen indexes. Of course, *k* must be different from *i*, to avoid that old and new location coincide, in order to find food sources having more nectar amount than the old one. The parameter ϕ_{ij} controls the production of neighbour food sources around x_{ij}, and represents the visual comparison of two food positions carried out by a bee. If a parameter value determined using Eq. (6) exceeds the constraints of the decision variables, the parameter is set to its upper and lower boundary, depending on which constraint has been exceeded.

As mentioned above, the food source abandoned by the bees is replaced with a new food source by the scouts at the third step of the cycle. In ABC algorithm, this is simulated by generating a random location and replacing the abandoned one with it. If a location cannot be further improved in a predetermined number of cycles, then that food source is assumed to be abandoned. The value of predetermined number of cycles, called ''limit", is an important control parameter of the algorithm. Karaboga and Akay (2009) proposed to determine this value as *SN*D*. This operation can be done using Eq. (7).

$$x_i^j = x_{\min}^j + \mathrm{rand}[0,1](x_{\max}^j - x_{\min}^j) \tag{7}$$

After each candidate source location v_{ij} is generated, its performance is compared with that of the old one. If the new food source has an equal or better nectar than the old source, it replaces the old one in the memory. Otherwise, the old one is retained in the memory. In other words, a greedy selection mechanism is employed as selection between the old and the candidate location. The steps of the ABC algorithm is given as:

Step 0: Initialize the population of solutions x_i, $i = 1,2,\ldots,SN$ and evaluate them
Step 1: Generate new solutions v_i for the employed bees by Eq. (6) and determine the quality of the solutions
Step 2: Apply the greedy selection process for the generated new solutions in Step 1.
Step 3: Calculate the probability values p_i for the solutions x_i by Eq. (5)
Step 4: Generate the new solutions for the onlookers from the solutions x_i due to the probabilities p_i using roulette wheel selection
Step 5: Apply the greedy selection process for the onlookers
Step 6: Determine the solution for the scout bee and replace it with produced solution x_i by Eq. (7)
Step 7: If the number of cycle is reached to *MCN*, the algorithm is terminated. Else go to Step 1.

5 Numerical Examples

In order to show the effectiveness and robustness of the ABC algorithm for CNDP, we have applied it on the test network adopted by Xu et al. (2009). It consists of 18 links and 6 nodes as shown in Fig. 1. The travel demand for this network includes three cases and is shown in Table 1. Case 1 describes the condition of light demand while case 2 and 3 are for heavy demand conditions. The results obtained through the ABC algorithm are compared with the results obtained by SA and GA taken from Xu et al. (2009) on the same network. Population size of the ABC algorithm (*SN*) and the maximum cycle number were set to 10 and 200, respectively.

Table 1. Travel demand cases for the example network

Case	q_{16}	q_{61}	Total flow
1	5	10	15
2	10	20	30
3	15	25	40

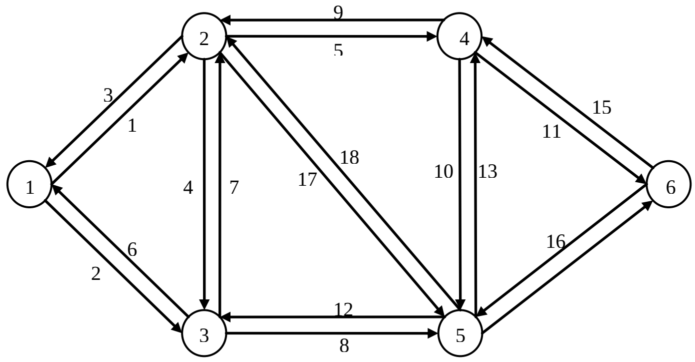

Figure 1. Example road network

The link travel time function is defined as shown in Eq. (8) and its corresponding parameters are given in Table 2.

$$t_a(x_a, y_a) = \alpha_a + \beta_a \left(\frac{x_a}{\theta_a + y_a}\right)^4 \quad (8)$$

The upper level objective function for the example road network is defined as in Eq. (9). u_a is set to the value of 20 for this example network, to compare the results with those obtained by Xu et al. (2009).

$$\min Z(x, y) = \sum_{a \in \boldsymbol{A}} (t_a(x_a, y_a) x_a + d_a y_a) \quad (9)$$

$$\text{s.t.} \qquad 0 \le y_a \le u_a \,, \quad \forall a \in \boldsymbol{A}$$

Table 2. Parameters for the eighteen link network

Link a	α_a	β_a	θ_a	d_a
1	1	10	3	2
2	2	5	10	3
3	3	3	9	5
4	4	20	4	4
5	5	50	3	9
6	2	20	2	1
7	1	10	1	4
8	1	1	10	3
9	2	8	45	2
10	3	3	3	5
11	9	2	2	6
12	4	10	6	8
13	4	25	44	5
14	2	33	20	3
15	5	5	1	6
16	6	1	4.6	1
17	5	9	45	2
18	5	9	45	2

The application of the ABC algorithm is tested on example road network, where the upper level optimization problem is solved using ABC algorithm, and UE traffic assignment is performed by way of FW method at the lower-level. The proposed algorithm has been executed in MATLAB programming, and performed on PC with Intel Core2 2.00 GHz, RAM 2 GB. Comparison of computation times for the ABC against the other compared algorithms for CNDP is difficult because each author uses a different computer, and in many cases, a different test network. However, one useful way of comparing computation times is in terms of the number of UE assignment (UE #) performed, since it is the most time consuming part of the algorithms. Therefore, this performance measure is used to compare the computation time of the ABC algorithm with the compared algorithms in this study.

In the initialization step, the ABC algorithm was initially structured with randomly generated solution vectors within its memory. At the first step of the cycle, employed bees come into the hive and share the quality of the solution vectors (objective function values) with the bees waiting on the dance area. At the second step, an onlooker chooses a food source area depending on the quality information of the solution vectors distributed by the employed bees on the dance area. At the third step of the cycle, when the nectar of a food source is abandoned by the bees, a new food source is randomly determined by a scout bee, and it replaces the abandoned one. These three steps are repeated until a predetermined *MCN* is reached. Then, it is assumed that the ABC algorithm has found optimum or near-optimum solution.

The results for cases 1, 2 and 3 on the example road network are given in Fig. 2. Table 3 shows comparison of the values of link capacity expansions obtained by ABC, SA and GA. The results show that the performance of the ABC is much better than the compared algorithms in terms of the number of UE assignment and the best objective function value.

Table 3. Comparison of ABC, SA and GA methods

	Case 1			Case 2			Case 3		
	ABC	SA	GA	ABC	SA	GA	ABC	SA	GA
y_1	0.00	0.00	0.00	0.69	0.00	0.00	3.09	0.00	0.00
y_2	0.00	0.47	0.00	1.66	1.73	2.20	6.76	9.12	11.98
y_3	0.00	0.65	0.00	9.83	11.77	10.61	13.39	18.12	16.24
y_4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_6	5.76	6.53	4.47	8.09	4.75	6.68	13.49	4.98	5.40
y_7	0.00	0.80	0.00	0.00	0.14	0.00	0.00	0.11	0.00
y_8	0.00	0.25	0.00	0.00	0.78	0.00	2.30	1.58	6.04
y_9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_{10}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_{11}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_{12}	0.00	0.00	0.00	0.00	0.00	0.00	2.52	0.00	0.00
y_{13}	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_{14}	0.00	0.84	0.00	1.32	5.94	1.22	12.08	11.66	12.28
y_{15}	0.04	0.14	0.00	0.01	1.51	6.30	0.15	2.97	0.82
y_{16}	7.88	7.34	7.54	19.99	18.45	11.93	19.99	19.71	19.99
y_{17}	0.24	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
y_{18}	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00
Z	190.40	205.89	191.26	488.50	505.39	515.09	735.09	739.54	744.39
UE #	2000	15000	50000	2000	42500	50000	2000	22500	50000

In case 1, the ABC initially started with random generated solutions and picked up the best objective function value of about 630, as can be seen in Fig. 2. The best objective function value, resulting after 2000 number of UE assignments (200 *MCN*), is found as 190.40 for same case using the ABC. SA and GA reach their optimal values of 205.89 and 191.26, after 15000 and 50000 number of UE assignments respectively. Thus, it is clear that, for this case, ABC is better than SA and GA in term of both number of UE assignments and best objective function value. It is remarkable that the ABC produced better functional value with much less number of UE assignments than the compared algorithms. Similarly, in case 2, the proposed algorithm yielded better performance in terms of both best objective function value and number of UE assignments. The ABC reached the value of 488.50 after 2000 UE assignments while SA and GA reach the values of 505.39 and 515.09, respectively, after 42500 and 50000 UE assigments, as shown in Table 3. In case 3, the ABC results slightly outperformed those generated by SA and GA, since they required much more number of UE assignments. The results showed that ABC is much more efficient and effective method than SA and GA for solving CNDP in terms of best objective function value and required number of UE assignments.

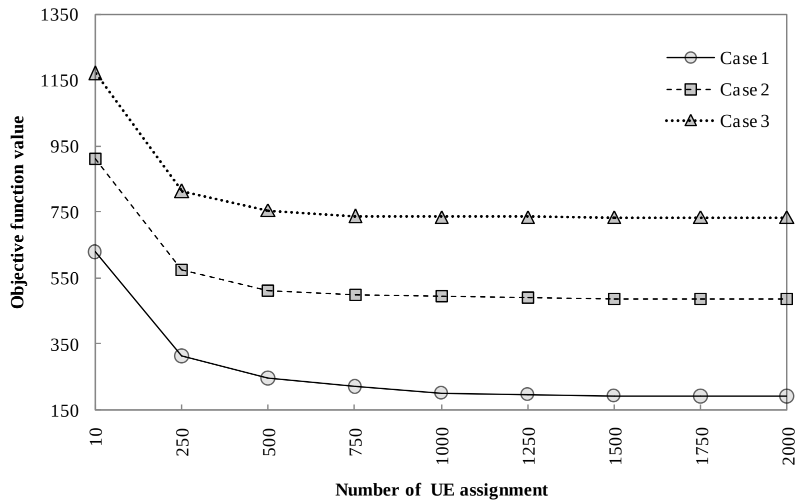

Figure 2. Performance of the ABC for all cases

6 Conclusions

In this paper, the ABC algorithm was proposed to solve CNDP with link capacity expansions. The CNDP is modeled as bi-level non-convex model. The upper level objective function is defined as the sum of total travel time and total investment costs of link capacity expansions on the road network, while the lower level problem is formulated as a user equilibrium traffic assignment model. To solve this problem we have used the Frank-Wolfe method. Numerical computations and comparisons have been carried out on example test network. According to the results, the ABC produced better results than SA and GA methods in terms of best objective function value and required number of UE assignments. The ABC outperformed SA and GA methods, and required much less number of UE assignments than the compared algorithms to reach the best functional values. It is notable that the ABC algorithm shows steady convergence towards the global or near global optimum for solving CNDP for all cases for the example network. In future work, the ABC will be applied to a real-sized network in order to demonstrate the applicability and the effectiveness of the proposed algorithm.

Acknowledgements

First author was supported by TUBITAK BIDEB 2219 program during this study at Technical University of Bari in Italy.

References

Abdulaal, M. and LeBlanc, L. (1979). Continuous equilibrium network design models. *Transportation Research Part B*, Vol. 13, pp. 19-32.

Ban, X.G., Liu, H.X., Lu, J.G. and Ferris, M.C. (2006). Decomposition scheme for continuous network design problem with asymmetric user equilibria. *Transportation Research Record (1964)*, pp. 185-192.

Chiou, S.W. (2005). Bilevel programming for the continuous transport network design problem. *Transportation Research Part B*, Vol. 39, pp. 361-383.

Davis, G.A. (1994). Exact local solution of the continuous network design problem via stochastic user equilibrium assignment. *Transportation Research Part B*, Vol. 28(1), pp. 61-75.

Fisk, C. (1984). Optimal signal controls on congested networks. *In: 9th International Symposium on Transportation and Traffic Theory*, VNU Science Press, pp. 197-216.

Frank, M. and Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics Quaterly*, Vol. 3, pp. 95-110.

Friesz, T.L., Anandalingam, G., Mehta, N.J., Nam, K., Shah, S.J. and Tobin, R.L. (1993). The multiobjective equilibrium network design problem revisited-A simulated annealing approach. *European Journal of Operational Research*, Vol. 65, No. 1, pp. 44-57.

Friesz, T.L., Cho, H.J., Mehta, N.J., Tobin, R.L. and Anandalingam, G. (1992). A simulated annealing approach to the network design problem with variational inequality constraints. *Transportation Science*, pp. 18-26.

Gao, Z., Sun, H. and Zhang, H. (2007). A globally convergent algorithm for transportation continuous network design problem. *Optimization and Engineering*, Vol. 8, pp. 241-257.

Karaboga, D. (2005). *An idea based on honeybee swarm for numerical optimization*. Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, Turkey.

Karaboga, D. and Akay, B. (2009). A comparative study of artificial bee colony algorithm. *Applied Mathematics and Computation*, Vol. 214, pp. 108-132.

Karaboga, D. and Basturk, B. (2007a). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization*, Vol. 39, pp. 459–471.

Karaboga, D. and Basturk, B. (2007b). Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems. *Lecture Notes in Artificial Intelligence 4529*, pp. 789–798, Springer-Verlag, Berlin.

Karaboga, D. and Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing*, Vol. 8, pp. 687–697.

Lucic, P. and Teodorovic, D. (2002). Transportation modeling: an artificial life approach. *in: ICTAI*, pp. 216–223.

Marcotte, P. and Marquis, G. (1992). Efficient implementation of heuristics for the continuous network design problem. *Annals of Operational Research*, Vol. 34, pp. 163-176.

Meng, Q., Yang, H., and Bell, M.G.H. (2001). An equivalent continuously differentiable model and a locally convergent algorithm for the continuous network design problem. *Transportation Research Part B*, Vol. 35, 83-105.

Sheffi, Y. (1985). *Urban Transport Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice-Hall Inc., New Jersey, USA.

Suwansirikul, C., Friesz, T.L. and Tobin, R.L. (1987). Equilibrium decomposed optimisation: a heuristic for the continuous equilibrium network design problem. *Transportation Science*, Vol. 21(4), pp. 254-263.

Teodorovic, D. (2003). Transport modeling by multi-agent systems: a swarm intelligence approach. *Transportation Planning and Technology*, Vol. 26(4), pp. 289-312.

Teodorovic, D. and Dell'Orco, M. (2005). Bee colony optimization–a cooperative learning approach to complex transportation problems, *10th EWGT Meeting*, in: Poznan, September 3-16, 2005.

Teodorovic, D. and Dell'Orco, M. (2008). Mitigating Traffic Congestion: Solving the Ride-Matching Problem by Bee Colony Optimization. *Transportation Planning and Technology*, Vol. 31(2), pp. 135-152.

Tereshko, V. (2000). *Reaction–diffusion model of a honeybee colony's foraging behaviour*. In: M. Schoenauer (Ed.), Parallel Problem Solving from Nature VI, Lecture Notes in Computer Science, Vol. 1917, pp. 807–816. Springer–Verlag, Berlin.

Xu, T., Wei, H. and Hu, G. (2009). Study on continuous network design problem using simulated annealing and genetic algorithm. *Expert Systems with Applications*, Vol. 36, pp. 1322-1328.