Available online at www.sciencedirect.com

ScienceDirect

Expert Systems with Applications 36 (2009) 1322–1328

Expert Systems with Applications

www.elsevier.com/locate/eswa

Study on continuous network design problem using simulated annealing and genetic algorithm

Tianze Xu [a,*], Heng Wei [b], Guanghua Hu [c]

[a] Department of Civil and Environmental Engineering, 735 Engineering Research Center, P.O. Box 210071, The University of Cincinnati, Cincinnati, OH 45221-0071, United States

[b] Department of Civil and Environmental Engineering, 792 Rhodes Hall, P.O. Box 210071, The University of Cincinnati, Cincinnati, OH 45221-0071, United States

[c] Department of Mathematics, Yunnan University, Kunming, China

Abstract

In general, a continuous network design problem (CNDP) is formulated as a bilevel program. The objective function at the upper level is defined as the total travel time on the network, plus total investment costs of link capacity expansions. The lower level problem is formulated as a certain traffic assignment model. It is well known that such bilevel program is nonconvex and algorithms for finding global optimal solutions are preferable to be used in solving it. Simulated annealing (SA) and genetic algorithm (GA) are two global methods and can then be used to determine the optimal solution of CNDP. Since the application of SA and GA on continuous network design on real transportation network requires solving traffic assignment model many times at each iteration of the algorithm, computation time needed is tremendous. It is important to compare the efficacy of the two methods and choose the more efficient one as reference method in practice. In this paper, the continuous network design problem has been studied using SA and GA on a simulated network. The lower level program is formulated as user equilibrium traffic assignment model and Frank–Wolf method is used to solve it. It is found that when demand is large, SA is more efficient than GA in solving CNDP, and much more computational effort is needed for GA to achieve the same optimal solution as SA. However, when demand is light, GA can reach a more optimal solution at the expense of more computation time. It is also found that increasing the iteration number at each temperature in SA does not necessarily improve solution. The finding in this example is different from [Karoonsoontawong, A., & Waller, S. T. (2006). Dynamic continuous network design problem – Linear bilevel programming and metaheuristic approaches. *Transportation Research Record (1964)*, 104–117, Network Modeling 2006.]. The reason might be the bi-level model in this example is nonlinear while the bi-level model in their study is linear.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Continuous network design; Traffic assignment; User equilibrium; Simulated annealing; Genetic algorithm

1. Introduction

The network design problem is to improve transportation network by selecting facilities (for example, entire-lane or new link) to add to a transportation network, or to determine capacity enhancements of existing facilities of a transportation network. Continuous network design problem (CNDP) is concerned with divisible capacity enhancements (Friesz, Cho, Mehta, Tobin, & Anandalingam, 1992), for example, altering lane width, median and shoulder area. In general, a continuous network design problem (CNDP) is formulated as a bi-level program. The upper level could be a multi-objective model or a model with objective function defined as the sum of total travel time on the network and total investment costs of link capacity expansions. The lower level problem is formulated as a certain traffic assignment model, which can be either a static traffic assignment model or dynamic traffic assignment model.

[*] Corresponding author. Tel.: +1 513 556 3128; fax: +1 513 556 2599.
E-mail address: xut@email.uc.edu (T. Xu).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.11.023

Determining the global optimal solution is of great importance in CNDP. It is well known that such a bi-level model is non-convex and non-differentiable. Global methods such as simulated annealing (SA) and genetic algorithm (GA) are preferable to be used in solving the model. Both SA and GA require evaluating the objective function value of the upper level model many times at each iteration of the algorithm. The lower level traffic assignment model has to be solved in order to evaluate the upper level objective function value. The computation time for solving a traffic assignment model of a real transportation network is considerably large. Thus the application of SA and GA to solve continuous network design requires tremendous computation time. It is important to compare the efficacy of the two methods and choose the more efficient one as reference method in practice.

In this paper, the continuous network design problem has been studied using SA and GA on a simulated network. The lower level program is formulated as static user equilibrium traffic assignment model and Frank–Wolf method is used to solve it. The computation time needed of each method for finding a optimal solution is compared.

2. Literature review

CNDP has been formulated as bi-level model of different kind. The upper level is formulated either as a multi-objective model (Friesz et al., 1993; Fan & Machemehl, 2006; etc.) or a model with objective function defined as the sum of total travel time on the network and total investment costs of link capacity expansions (Friesz et al., 1992; Chiou, 2005; Karoonsoontawong & Waller, 2006; etc.). The lower level problem is formulated as a certain traffic assignment model, such as static user equilibrium (Friesz et al., 1992; Chiou, 2005; Ban, Liu, Lu, & Ferris, 2006), variable demand equilibrium (Chen & Chou, 2006), stochastic user equilibrium (Davis, 1994; Chen, Subprasom, & Ji, 2006), user-optimal dynamic traffic assignment (UO DTA) model (Karoonsoontawong & Waller, 2006), etc.

Different methods have been used to solve CNDP models. Study by Friesz et al. (1992) shows that simulated annealing (SA) is superior to Iterative Optimization-Assignment algorithm (IOA), Hooke-Jeeves algorithm (HJ), Equilibrium Decomposed Optimization (EDO), and Modular In-core Nonlinear System (MINOS) in finding global optimal solution when the lower model is formulated as static user equilibrium. Davis (1994) used the generalized reduced gradient method and sequential quadratic programming to find the optimal solution of CNDP. Chiou (2005) proposed four gradient-based methods to solve the CNDP. The four gradient-based methods are Gradient Projection method (GP), Conjugate Gradient projection method (CG), Qusai-NEWton projection method (QNEW), and PARATAN version of gradient projection method (PT). He applied the methods to the sixteen link network used by Friesz et al. (1992) and Sioux Falls city network and compared the four methods with other methods including IOA, HJ, EDO, MINOS, SA, Sensitivity Analysis Based algorithm (SAB), and Augmented Lagrangian algorithm (AL). The comparison shows that CG and QNEW outperforms all other methods except SA. SA outperforms all other methods in all cases. But the difference between CG, QNEW and SA is not large. Ban, Liu, Ferris, and Ran (2006) presented a Relaxation method (RELAX) to solve CNDP when the lower level is a nonlinear complementary problem. They also applied the methods to the same network as Chiou's study and compared RELAX with other methods including IOA, EDO, MINOS, SA, and AL. Their study shows SA and RELAX outperforms all other methods. In total three cases, SA outperforms RELAX once and RELAX outperforms SA twice. But the difference is very small. Karoonsoontawong and Waller (2006) used simulated annealing (SA), genetic algorithm (GA), and random search (RS) to find optimal solutions of CNDP when it is modeled as a linear bi-level programming with a lower level of dynamic user optimal traffic assignment model. His study shows that GA outperforms the others in the test problems.

The objective of this paper is to study continuous network design problem using SA and GA when the lower level is modeled as static user equilibrium (UE). Specifically, the efficiency of the two global methods will be compared. Though Karoonsoontawong and Waller's (2006) study has showed that GA outperforms SA when the lower level is dynamic traffic assignment, it is still necessary to compare the efficacy of SA and GA when the lower level is modeled as static traffic assignment because the bi-level static model is non-convex nonlinear and is different from linear bi-level programming in their study. Since the time needed for solving the lower traffic assignment model accounts for major proportion of computation time of the whole problem, the UE assignment number will be used to measure the computation cost. It is more reasonable to use UE assignment number instead of CPU time to measure computation cost of either method since it is independent of network and is more comparable.

3. Problem formulation

The continuous network design problem (CNDP) with fixed demand static user equilibrium flow constraint is formulated as

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

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

where $x(y)$ is the equilibrium flow defined by the following fixed demand static user equilibrium problem:

$$\min \quad z(\boldsymbol{x}) = \sum_a \int_0^{x_a} t_a(\omega, y_a)\,\mathrm{d}\omega \tag{2}$$

$$\text{s.t.} \quad \sum_l f_l^{rs} = q_{rs} \quad \forall r \in \boldsymbol{R}, s \in \boldsymbol{S}$$

$$x_a = \sum_{rs} \sum_{k \in K_{rs}} f_k^{rs} \delta_{a,k}^{rs} \quad \forall r \in \boldsymbol{R}, s \in \boldsymbol{S}, a \in \boldsymbol{A}, k \in \boldsymbol{K_{rs}}$$

$$f_k^{rs} \geqslant 0 \quad \forall r \in \boldsymbol{R}, s \in \boldsymbol{S}, k \in \boldsymbol{K_{rs}}$$

and where $\boldsymbol{A}$ is the set of links in the network; $\boldsymbol{R}$ is the set of origins; $\boldsymbol{S}$ is the set of destinations; $\boldsymbol{D}$ is the vector of fixed OD pair demands, $\boldsymbol{D} = [\boldsymbol{D}_{rs}]$ $\forall r \in \boldsymbol{R}$, $s \in \boldsymbol{S}$; $\boldsymbol{K}_{rs}$ is the set of paths between OD pair rs $\forall r \in \boldsymbol{R}$, $s \in \boldsymbol{S}$; $\boldsymbol{f}$ is the vector of path flows, $\boldsymbol{f} = [f_k^{rs}]$, $\forall r \in \boldsymbol{R}$, $s \in \boldsymbol{S}$, $k \in \boldsymbol{K}_{rs}$; $\boldsymbol{x}$ is the vector of equilibrium link flows, $\boldsymbol{x} = [x_a]$ $\forall a \in \boldsymbol{A}$; $\boldsymbol{y}$ is the vector of link capacity expansions, $\boldsymbol{y} = [y_a]$ $\forall a \in \boldsymbol{A}$; $\boldsymbol{u}$ is the vector of upper bound for link capacity expansions, $\boldsymbol{u} = [u_a]$ $\forall a \in \boldsymbol{A}$; ρ is the conversion factor from investment cost to travel times; $\boldsymbol{t}$ is the vector of link travel times, $\boldsymbol{t} = [t_a(x_a, y_a)]$ $\forall a \in \boldsymbol{A}$; $\boldsymbol{g}$ is the vector of investment costs, $\boldsymbol{g} = [g_a(y_a)]$ $\forall a \in \boldsymbol{A}$; Δ is the link-path incidence matrix, $\Delta = [\delta_{a,k}^{rs}]$, where $\delta_{a,k}^{rs} = 1$ if link a is on the kth route connecting origin r and destination s, and $\delta_{a,k}^{rs} = 0$ otherwise.

In this bi-level model, the upper level is non-convex and non-differentiable in which is defined by the lower level model. Below we will introduce two global methods including SA and GA to solve the bi-level model. The efficacy of the two methods will be compared.

4. Genetic algorithm (GA)

A genetic algorithm (GA) is a global search heuristic technique used in computing to find true or approximate solutions to optimization problems. It uses techniques such as inheritance, mutation, selection, and crossover which are inspired by evolutionary biology. Genetic algorithms are implemented as a computer simulation in which a population of chromosomes of candidate solutions to an optimization problem evolves toward better solutions. Solutions can be represented in binary or real coded. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness value of every individual in the population is evaluated, multiple individuals are randomly selected from the current population (based on their fitness value), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

A real-coded genetic algorithm with multiple crossovers was used in this paper. The algorithm is well described by Chang (2007). A brief summary of the algorithm is given below for completeness.

4.1. Initialization

A search space should be first defined. All genes in the chromosome will be operated and evaluated in this constrained space. Let population size N represent the number of original chromosomes. The initial solution was generated randomly in the feasible region. Once a generated chromosome by genetic operations goes beyond the bound, the original chromosome will be retained.

4.2. Reproduction

A simpler tournament selection is adopted to decide whether a chromosome can reproduce or not based on its fitness value (z). The tournament selection says that $p_{\mathrm{r}}N$ chromosomes with minimum z values are more added into the population and $p_{\mathrm{r}}N$ chromosomes with maximum z values are discarded from the population, where p_{r} is the probability of reproduction. The resulting population has the same size with the original one. After the selection, all chromosomes are completely put in the mating pool.

4.3. Crossover

The N chromosomes are randomly divided into $N/2$ pairs. Assume that both $\boldsymbol{y}$ and $\bar{\boldsymbol{y}}$ are selected and c is a random number chosen from $[0,1]$. If $c > p_{\mathrm{c}}$, where p_{c} is probability of crossover, then the following crossover operations for $\boldsymbol{y}$ and $\bar{\boldsymbol{y}}$ are performed

$$\begin{aligned}
&\text{if } z(\boldsymbol{y}) < z(\bar{\boldsymbol{y}}) \\
&\qquad \boldsymbol{y}' = \boldsymbol{y} + r(\boldsymbol{y} - \bar{\boldsymbol{y}}), \\
&\qquad \bar{\boldsymbol{y}}' = \bar{\boldsymbol{y}} + r(\boldsymbol{y} - \bar{\boldsymbol{y}}), \\
&\text{else} \\
&\qquad \boldsymbol{y}' = \boldsymbol{y} + r(\bar{\boldsymbol{y}} - \boldsymbol{y}), \\
&\qquad \bar{\boldsymbol{y}}' = \bar{\boldsymbol{y}} + r(\bar{\boldsymbol{y}} - \boldsymbol{y}),
\end{aligned} \tag{3}$$

where $z(\boldsymbol{y})$ and $z(\bar{\boldsymbol{y}})$ are fitness values of chromosomes $\boldsymbol{y}$ and $\bar{\boldsymbol{y}}$, respectively, $\boldsymbol{y}'$ and $\bar{\boldsymbol{y}}\prime$ are the resulted children chromosomes, and $r \in [0,1]$ is a random number determining the crossover grade of these two. If $c < p_{\mathrm{c}}$, no crossover operation is performed.

4.4. Mutation

The mutation operation follows the multiple crossover and provides a possible mutation on some selected chromosomes. Only $p_{\mathrm{m}}N$ random chromosomes in the current population are chosen to be mutated. The formula of mutation operation for a selected $\boldsymbol{y}$ is given by

$$\boldsymbol{y}' = \boldsymbol{y} + s \cdot \theta \tag{4}$$

where s is a small positive constant and θ is a random perturbation vector to produce small disturbances on $\boldsymbol{y}$.

The overall real-coded GA on CNDP is summarized as follows:

```
Step 1: Create a population with $N$ chromosomes, which are randomly generated from the feasible region.
Step 2: For each chromosome in the population, solve program (2) and evaluate the fitness value with (1).
Step 3: If the pre-specified number of generations is reached or there is a chromosome in population with fitness value less than prespecified error, stop; else, continue.
Step 4: Reproduction.
Step 5: Perform multiple crossover based on (3).
Step 6: Perform mutation based on (4). (If the resulting chromosome during these operations is outside the region, the original one is retained).
Step 7: Go back to Step 2.
```

5. Simulated annealing (SA)

Simulated annealing (SA) is a stochastic gradient method for the global optimization problem. It was originally developed to simulate the annealing process. It starts from an initial solution at a high temperature, and makes a series of moves according to annealing schedule. The change in the objective function values (ΔE) is computed at each move. If the new solution results in decreased objective function value, it is accepted with probability 1. If the new solution yields increased objective function value, it is accepted with a small probability p, which is defined as $P(\Delta E) = \exp(-\Delta E/k_B T)$, where k_B is Boltzmann's constant and T is the current temperature. By accepting worse solutions with a certain probability, SA can avoid being trapped on a local optimum. SA repeats this process M times at each temperature to reach the thermal equilibrium, where M is a control parameter, also known as Markov length (Wu et al., 2007). The parameter T is gradually decreased as SA proceeds until the stopping condition is met. It terminates, when either the optimal solution is attained or the problem becomes frozen at a local optimum that cannot be improved.

The algorithm is well described by Liu and Canqi (2001). A brief summary of the algorithm is given below for completeness.

Some factors need to be considered when designing the SA algorithm are introduced first.

Initialization: An initial solution is generated randomly from the feasible region. An initial temperature should be high enough to allow all candidate solutions to be accepted.

Markov length: The iteration number M used in each temperature. This number should be set appropriately high for the objective function values to reach Boltzman distribution.

Cooling schedule: Cooling schedule is the rate at which the temperature is reduced. In this paper, 0.8 is used at the first 12 temperature reductions. 0.8 means the temperature of the next stage is 0.8 times the current temperature. 0.5 is used after the 12th temperature reduction.

Step size: Step size at each move should be decreased with the reduction of temperature. The feasible solutions at lower temperature are close to optimal solution. When temperature is low, the stochastic search tends to be deterministic search. So if step size is too large, at low temperature, some feasible solutions will be rejected, thus computation time will be wasted.

Neighboring solutions: Neighboring solutions are the set of feasible solutions that can be generated from the current solution. Each feasible solution can be directly reached from current solution by a move and resulted neighboring solution.

Stopping criteria: The algorithm stops when the number of temperature transitions reaches a prespecified number, or when the temperature is reduced to a threshold, or when the neighbor solution was not improved after a period.

The algorithm of SA on CNDP is summarized as:

```
Step 1: Initialization.
    1.1 Generate an initial feasible solution $\mathbf{y}^0$, solve program (2) and evaluate the objective function value $\mathbf{z}(\mathbf{y}^0)$ with (1) based on $\mathbf{y}^0$, let $\mathbf{y} = \mathbf{y}^0$.
    1.2 Set Markov length $M$, initial step size $\alpha_0$, initial temperature $T_0$, error $\varepsilon$, let
        $\alpha = \alpha_0, T = T_0$.
    1.3 Set the outer iteration counter $n = 1$.
Step 2: For the given $T$, perform the following:
    2.1 Set the inner iteration counter $k = 1$.
    2.2 Let $\hat{\mathbf{y}} = \mathbf{y} + \alpha\mathbf{U}, n = n + 1$, where $\mathbf{U}=(\ldots,U_j,\ldots)$ is a random vector and $U_j$ is independently uniformly distributed on $[-\sqrt{3}, \sqrt{3}]$.
    2.3 Solve program (2) and evaluate the objective function value $\mathbf{z}(\hat{\mathbf{y}})$ with (1) based on $\hat{\mathbf{y}}$.
    2.4 Set $\Delta\mathbf{z} = \mathbf{z}(\hat{\mathbf{y}}) - \mathbf{z}(\mathbf{y})$, if $\Delta\mathbf{z} < 0$, $\mathbf{y} = \hat{\mathbf{y}}$; else let $\mathbf{y} = \hat{\mathbf{y}}$ with probability $P(\Delta\mathbf{z})=\exp(-\Delta\mathbf{z}/T)$.
    2.5 $k = M$? If $k = M$, go to step 3; else $k = k + 1$, return to 2.2.
Step 3: Stop or not
    3.1 $T < \varepsilon$? If $T < \varepsilon$, stop; else set $\alpha = \frac{n-1}{n}\alpha$.
    3.2 $n \geqslant 12$? If $n \geqslant 12$, let $T = 0.5T$, $n = n + 1$; else $T = 0.8\ T$, $n = n + 1$, return to Step 2.
```

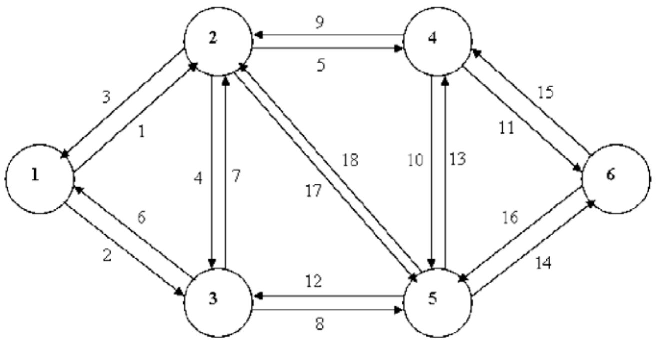

Fig. 1. Eighteen link network.

Table 1
Travel demand for the example

Case	Travel demand from 1 to 6 q_{16}	Travel demand from 6 to 1 q_{61}	Total flow
1	5	10	15
2	10	20	30
3	15	25	40

Table 2
Network parameters for the example

Arc a	A_a	B_a	K_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

Table 3
Parameters for simulated annealing algorithm in the example

Parameter	Parameter value
Initial temperature	500
Final temperature	0.0052
Temperature reduction number	25
Iterations at each temperature M	300,400,500,...,1800,1900,2000
Step at each iteration	$\alpha_n = \frac{n-1}{n}\alpha_{n-1}$
Upper bound on y_a	20

Table 4
Parameters for genetic algorithm in the example

Parameter	Parameter value
Population size N	10
Probability of reproduction p_r	0.2
Probability of crossover p_c	0.2
Probability of mutation p_m	0.2
Step in mutation operation s	0.5
Upper bound on y_a	20
Number of generations	750,1000,1250,...,4500,4750,5000

6. Numerical example

In this section, the efficacy of the GA and SA for continuous network design programs of the simulated network is described and compared. The network is shown in Fig. 1. It consists of 18 links and 6 nodes. This network is a modified network from the network used by Friesz et al. (1992). The travel demand for this network includes three travel demand cases and is shown in Table 1. Case 1 is for light demand. Case 2 and case 3 are for heavy demand. The parameters for the network are shown in Table 2. The travel time function of each link is defined as $t_a(x_a, y_a) = A_a + B_a(x_a/(K_a + y_a))^4$. The parameters used in SA are summarized in Table 3. The parameters used in GA are summarized in Table 4.

The comparison of SA and GA are shown in Table 5. In case 1 in which demand is light, SA reaches an optimal value of 205.8907 and 15000 UE assignments need to be

Table 5
Comparison of SA and GA

	Case 1		Case 2		Case 3	
	SA	GA	SA	GA	SA	GA
y_1	0	0	0	0	0	0
y_2	0.4688	0	1.7313	2.2047	9.1244	11.9861
y_3	0.6543	0	11.7700	10.6133	18.1220	16.2407
y_4	0	0	0	0	0	0
y_5	0	0	0	0	0	0
y_6	6.5331	4.4698	4.7528	6.6830	4.9850	5.4073
y_7	0.7965	0	0.1415	0	0.1127	0
y_8	0.2460	0	0.7823	0.0026	1.5768	6.0434
y_9	0	0	0	0	0	0
y_{10}	0	0	0	0	0	0
y_{11}	0	0	0	0	0	0
y_{12}	0	0	0	0	0	0
y_{13}	0	0	0	0	0	0
y_{14}	0.8421	0	5.9448	1.2167	11.6649	12.2779
y_{15}	0.1361	0	1.5119	6.3021	2.9716	0.8197
y_{16}	7.3447	7.5403	18.4496	11.9289	19.7191	19.9897
y_{17}	0	0	0	0	0	0
y_{18}	0	0	0	0	0	0
z^*	205.8907	191.2557	505.3919	515.0984	739.5390	744.3962
UE #	15,000	50,000	42,500	50,000	22,500	50,000

made to reach this optimal; GA reaches an optimal value of 191.2557 and 50000 UE assignments need to be made to reach this optimal. In case 2 and case 3, SA reach optimal value of 505.3919 and 739.5390 and the corresponding UE assignment number is 42500 and 22500; GA reach optimal value of 515.0984 and 744.3962 and 50000 UE assignments are needed.

The comparison is made clearer by plotting the optimal objective function value versus UE assignment number (UE#) needed to reach the optimal for both SA and GA. The plot is shown in Figs. 2–4, which are for case 1, case 2 and case 3, respectively. Some observations can be made based on the plot. When demand is light as in case 1, GA can reach a more optimal value than SA at the expense of more UE assignments. When demand is heavy as in case 2 and case 3, the optimal values found by SA and GA are about the same, but the UE assignment number for GA is much more than that in SA. In three cases, the objective function value by GA decreases consistently with the increase of UE assignment number; however, solutions by SA do not show obvious improvement with the increase of UE assignment number (which is made by increasing Markov length or the iteration number at each temperature). In SA, solution found with lower Markov length (for example 300) is as optimal as solutions found with higher Markov length (for example 1800). This shows that increasing the iteration number at each temperature in SA does not necessarily improve solution and an appropriately low Markov length should be used to save computation time. It is interesting to note the performance of SA and GA in this example is different from the result in Karoonsoontawong and Waller's study (2006) because the bi-level model in this example is nonlinear while the bi-level model in their study is linear.

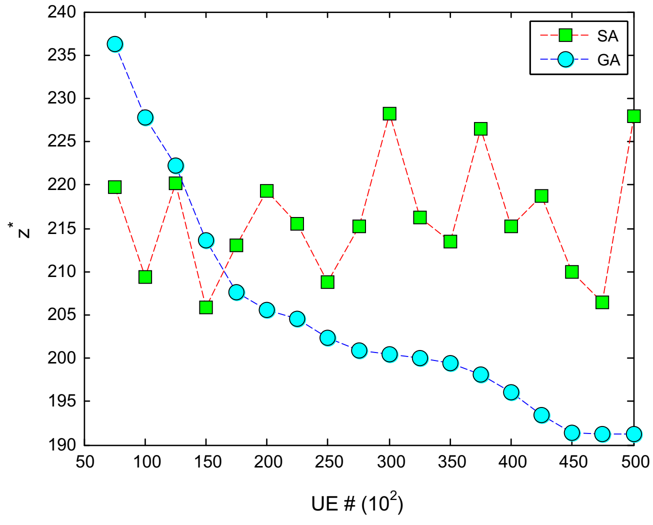

Fig. 2. Plot of z^* versus UE # for case 1.

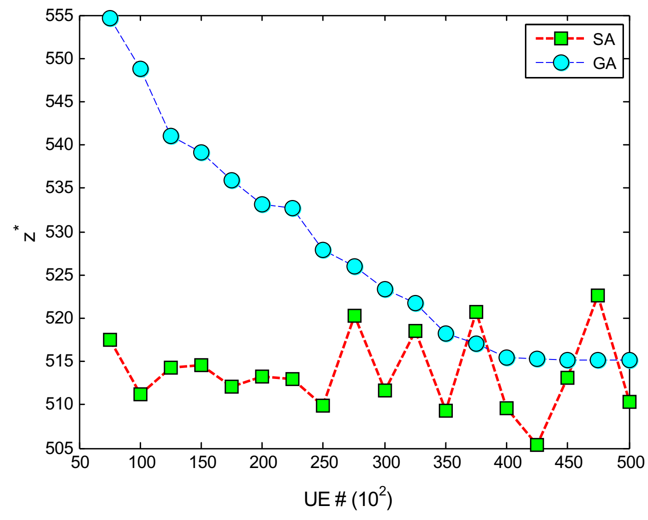

Fig. 3. Plot of z^* versus UE # for case 2.

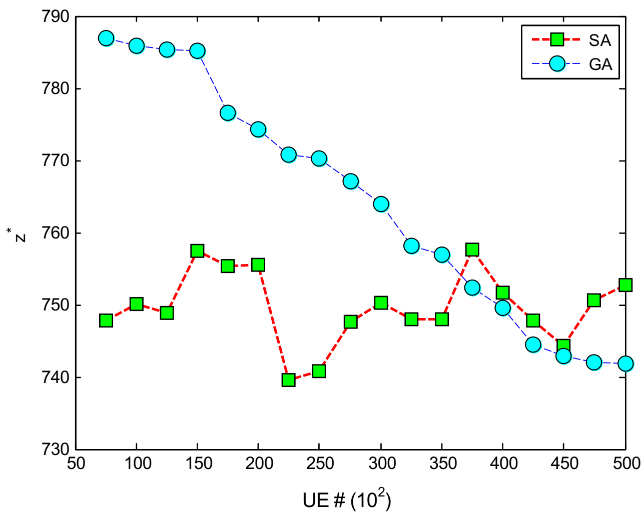

Fig. 4. Plot of z^* versus UE # for case 3.

7. Conclusion

In this paper, the continuous network design problem (CNDP) has been studied using SA and GA on a simulated network. The CNDP is modeled as a bi-level non-convex nonlinear program. The objective function at the upper level is defined as the total travel time on the network, plus total investment costs of link capacity expansions. The lower level program is formulated as user equilibrium traffic assignment model. Two global methods including SA and GA are used to find the optimal solution of the CNDP program and the efficacy of the two methods are compared. It is found that when demand is light, SA is more efficient than GA in solving CNDP and much more computational effort is needed for GA to find the same optimal solution as SA. However, when demand is light, GA can reach a more optimal solution at the expense of more computation time. It is also found that increasing the iteration number at each temperature in SA does not necessarily improve solution. Our finding is different from Karoonsoontawong and

Waller's study (2006). The reason might be the bi-level model in this example is nonlinear while the bi-level model in their study is linear.

References

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

Ban, J. X., Liu, H. X., Ferris, M. C., & Ran, B. (2006). A general MPCC model and its solution algorithm for continuous network design problem. *Mathematical and Computer Modelling, 43*(5–6), 493–505.

Chang, Wei-Der (2007). A multi-crossover genetic approach to multivariable PID controllers tuning. *Expert Systems with Applications, 33*, 620–626.

Chen, A., Subprasom, K., & Ji, Z. W. (2006). A simulation-based multi-objective genetic algorithm (SMOGA) procedure for BOT network design problem. *Optimization and Engineering, 7*(3), 225–247.

Chen, H. K., & Chou, H. W. (2006). Reverse supply chain network design problem. *Transportation Research Record (1964)*, 42–49, Network Modeling 2006.

Chiou, S. W. (2005). Bilevel programming for the continuous transport network design problem. *Transportation Research Part B-Methodological, 39*(4), 361–383.

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

Fan, W., & Machemehl, R. B. (2006). Optimal transit route network design problem with variable transit demand: Genetic algorithm approach. *Journal of Transportation Engineering-ASCE, 132*(1), 40–51.

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

Friesz, T. L., Anandalingam, G., Mehta, N. J., Nam, K., Shah, S. J., & Tobin, R. L. (1993). The multiobjective equilibrium network design problem revisited – A simulated annealing approach. *European Journal of Operational Research, 65*(1), 44–57.

Karoonsoontawong, A., & Waller, S. T. (2006). Dynamic continuous network design problem – Linear bilevel programming and metaheuristic approaches. *Transportation Research Record (1964)*, 104–117, Network Modeling 2006.

Liu & Canqi (2001). *Advanced traffic planning*. People's Transportation Press (pp. 345–346).

Wu, Tai-Hsi, Chang, Chin-Chih, & Chung, Shu-Hsing (2007). A simulated annealing algorithm for manufacturing cell formation problems. *Expert Systems with Applications, 34*(3), 1609–1617.