JOURNAL OF TRANSPORTATION SYSTEMS ENGINEERING AND INFORMATION TECHNOLOGY
Volume 9, Issue 3, June 2009
Online English edition of the Chinese language journal

Available online at www.sciencedirect.com
ScienceDirect

Cite this article as: J Transpn Sys Eng & IT, 2009, 9(3), 64−70.

RESEARCH PAPER

Sensitivity Analysis of Simulated Annealing for Continuous Network Design Problems

YANG Jin, XU Meng*, GAO Ziyou

School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China

Abstract: In this paper, parameters choices of simulated annealing for continuous network design problems are discussed. A bi-level programming model for continuous network design problem is introduced. Objective function of the upper level is defined as the sum of the total travel time on the network and the total investment costs of link capacity expansions. The lower level problem is the user equilibrium assignment model, which is solved by the Gradient projection algorithm. Sensitivity analysis method is the first time used to analyze and compare the influence of the different selection of parameters to the implementation of simulated annealing algorithm. Suggestions of parameter selection are also given. Analysis demonstrates that the efficiency and precision of these methods can be improved clearly with the proposed suggestions.

Key Words: continuous network design problem; bi-level programming model; simulated annealing; sensitivity analysis

1 Introduction

Continuous network design problem (CNDP) is concerned with the optimal capacity expansion of existing links in a given network by minimizing the total system cost as well as considering the route choice behavior of individual users. Due to multiple objectives for formulating CNDP, it was modeled as a bilevel programming problem, where the upper level is a nonlinear programming problem to minimize the system cost and the lower level is user equilibrium (UE) problem to assume drivers' route choice behavior[1]. In the transportation area, the bilevel programming models of CNDP was first proposed by Abdulaal and LeBlanc[2]. There are many researchers focusing on the studies of CNDP. Early detailed reviews and improved models can be found in Gao *et al*.[1,3–5].

How to design efficient algorithms for solving the CNDP formulated as the bilevel model is an important work, and fascinates practitioners' interest[6]. In this paper, we have focused on the algorithm implementation issues with simulated annealing to solve CNDP. Some early work on this topic can be referred to Friesz *et al*.[7,8]. Recently, both Gui[9] and Liu[10] studied the algorithm design problems of CNDP solved by simulated annealing, and the difference lies in using different algorithms for solving the UE on the lower problem. Xu *et al*.[11] compared the differences to solve CNDP with genetic algorithm and simulated annealing algorithm under different demand, and concluded that the implementation efficiency of the simulated annealing algorithm is higher than genetic algorithm under low demand, and it needs more operations with genetic algorithm to arrive at the same precision level.

When the simulated annealing algorithm is used to solve CNDP, the setting of parameters have clear effects on the solution and solving time, for example, the choice methods and choosing rulers of the inner iteration number, initial temperature, the temperature decrease rate, and the lowest temperature, and so on. It is a difficult problem to tell how to combine these parameters in order to get the best performance of the algorithm. Up to now, there is no theoretical method to approach this problem, and the setting of the parameters depends on experience or trial. In this paper, the parameter setting problems is first discussed with sensitivity analysis methods when using simulated annealing algorithm to solve CNDP, which could provide useful advice for the parameters setting.

Received date: Nov 6, 2008; Revised date: Apr 13, 2009; Accepted date: May 7, 2009
*Corresponding author. E-mail: mengxu@bjtu.edu.cn

DOI: 10.1016/S1570-6672(08)60064-1

2 Bilevel programming formulation for CNDP

This section provides a summary of bilevel programming formulation for CNDP, and two sensitivity analysis methods including one-at-a-time designs (OATD) and factorial designs (FD). The presentation in this summary section follows Gao *et al*.[1,12]. Notation is provided firstly for convenience, followed by the bilevel programming formulation and OATD and FD methods.

2.1 Notation

Considering a transportation network $\boldsymbol{G}(\boldsymbol{N}, \boldsymbol{A})$, the following notations are given:

$\boldsymbol{A}$—set of links, $\forall a \in \boldsymbol{A}$;

$\boldsymbol{R}$—set of origins, $\forall r \in \boldsymbol{R}$;

$\boldsymbol{S}$—set of destinations, $\forall s \in \boldsymbol{S}$;

$\boldsymbol{P}_{rs}$—set of routes between OD pair (r, s), $\forall r \in \boldsymbol{R}$, $\forall s \in \boldsymbol{S}$;

x_a—flow on link $a, \forall a \in \boldsymbol{A}$, $\boldsymbol{x} = (\cdots, x_a, \cdots)$;

f_p^{rs}—flow on route p between OD pair (r, s), $p \in \boldsymbol{P}_{rs}$, $\boldsymbol{f} = (\cdots, f_p^{rs}, \cdots)$;

q_{rs}—travel demand between OD pair (r, s), $q = (q_{rs})_{|R|\times|S|}$;

$\delta_{a,p}^{rs}$—indicator variable: if link a is on route p between OD pair (r, s), $\delta_{a,p}^{rs} = 1$; $\delta_{a,p}^{rs} = 0$, otherwise. $\left(\Delta^{rs}\right)_{a,p} = \delta_{a,p}^{rs}$, $\boldsymbol{\Delta} = \left(\cdots, \Delta^{rs}, \cdots\right)$;

y_a—capacity expansion on link $a \in \boldsymbol{A}$, $\boldsymbol{y} = (\cdots, y_a, \cdots)$, $y_a \in \left[\underline{y}_a, \overline{y}_a\right]$;

$G_a(y_a)$—investment costs function on link a, $\boldsymbol{G}(\boldsymbol{y}) = \left(\cdots, G_a\left(y_a\right), \cdots\right)$;

$t_a(x_a, y_a)$—travel time on link $a \in \boldsymbol{A}$, $\boldsymbol{t}(\boldsymbol{x}, \boldsymbol{y}) = \left(\cdots, t_a\left(x_a, y_a\right), \cdots\right)$;

ϕ—conversion factor from investment cost to travel cost.

2.2 Problem description

Thus CNDP can be formulated in terms of the bi-level programming model as follows[1]:

(P1) (L1)

$$\min z(\mathbf{x}, \mathbf{y}) = \sum_{a \in \mathbf{A}} \int_0^{x_a(\mathbf{y})} t_a(v, y_a)\, \mathrm{d}v \tag{1}$$

$$\text{s.t.} \quad \sum_{p \in \boldsymbol{P}_{rs}} f_p^{rs} = q_{rs}, \forall r \in \boldsymbol{R}, s \in \boldsymbol{S} \tag{2}$$

$$f_p^{rs} \geq 0, \forall r \in \boldsymbol{R}, s \in \boldsymbol{S}, p \in \boldsymbol{P}_{rs} \tag{3}$$

$$x_a = \sum_r \sum_s \sum_p f_p^{rs} \delta_{a,p}^{rs}, \forall a \in \boldsymbol{A} \tag{4}$$

(P1) (U1)

$$\min Z = \sum_{a \in \boldsymbol{A}} t_a(x_a(\boldsymbol{y}), y_a) x_a(\boldsymbol{y}) + \phi \sum_{a \in \boldsymbol{A}} G_a(y_a) \tag{5}$$

$$\text{s.t.} \quad \underline{y}_a \leq y_a \leq \overline{y}_a, \forall a \in \boldsymbol{A} \tag{6}$$

Equation (5) is the objective function of CNDP, which minimizes the whole network cost and investment cost by improving some links' ability. In this paper, the ϕ can set 1 or 1.5, and $G_a(y_a)=d_a \cdot y_a$ or $G_a(y_a)=d_a \cdot (y_a)^2$. Expression (6) requires the improved ability that is nonnegative and satisfies the constraints of upper and lower bounds.

3 Algorithm

In this paper, the gradient projection (GP) was used to solve the UE model (L1). GP is a typical path-based algorithm, and the convergence is faster than the Frank-Wolfe algorithm[13].

The overall process with simulated annealing algorithm to solve CNDP (P1) (U1) is summarized as follows:

Step 1: initialization. Given randomly the decision variable an initial solution y^0 in its field, and let $y=y^0$, it can get the value of $\boldsymbol{x}$ and corresponding objective function value $Z(x, y)$ by solving the lower UE model. Set the inner iteration number M, the initial temperature value T_0, the lowest temperature ε and set $T=T_0$, the temperature decrease rate α, and initial stepsize l.

Step 2: set the inner iteration number $k=0$. And set $\hat{\boldsymbol{y}} = \boldsymbol{y} + l \cdot \boldsymbol{U}$, where $\boldsymbol{U}$ is a stochastic vector, $\boldsymbol{U} = (\cdots, U_i, \cdots)$, and U_i is a random variable and follows the uniform distribution on [–1, 1]. Solve the lower UE model, get the value $\hat{\boldsymbol{x}}$, and calculate the objective function value $Z(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$.

Step 3: if $\Delta Z = Z(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}) - Z(\boldsymbol{x}, \boldsymbol{y}) < 0$, let $\boldsymbol{y} = \hat{\boldsymbol{y}}$; or get the value $\hat{\boldsymbol{y}}$ with the possibility $p = \exp[(-\Delta Z)/T]$, that is, generate a random value r on [0, 1], set $\boldsymbol{y} = \hat{\boldsymbol{y}}$ if $r<p$.

Step 4: If $k=M$, go to Step 5, otherwise, set $k=k+1$ and return to Step 2.

Step 5: If $T<\varepsilon$, algorithm stop and output decision variable y and objective function value Z; otherwise, update stepsize l, and current temperature $T=\alpha \cdot T$, and return Step 2.

In the above algorithm, the initial stepsize l depends on the practical problem, and the updating of stepsize can depend on $l=l \cdot \beta$, where β is the stepsize decrease ratio. Assume l_0 is the maximization stepsize, and l_f is the minimization stepsize, it can set $\beta = \sqrt[n]{l_g / l_0}$, where $n=(\ln\varepsilon - \ln T_0)/\ln\alpha$, this method can reduce the number of the parameters. Furthermore, to avoid the small changes of decision variable y to solve the lower model (which add the cost of implementation time), it is set that passing over to solve the lower problem with the small changes of y.

4 Sensitivity analysis

In the contexts of numerical modeling, sensitivity analysis (SA) study the relationships between information flowing in and out of model. SA is widely used in model development, verification, calibration, model identification and mechanism reduction. It can assist the modeler to determine whether the parameters are sufficiently precise for the model to give reliable predictions. In this paper, the sensitivity analysis (SA) is introduced to appraise the estimated parameters of simulated annealing to the CNDP. Two methods (one-at-a-time designs (OATD) and factorial designs (FD)) are used to analyze the effects of parameters[12].

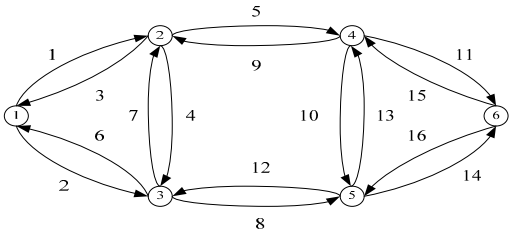

Fig. 1 Test network

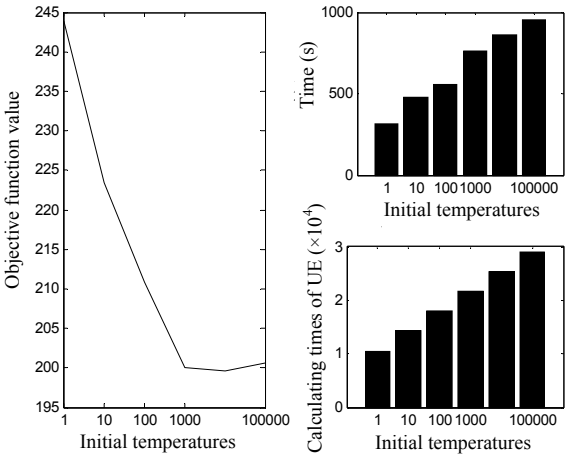

Fig. 2 Results in different initial temperatures

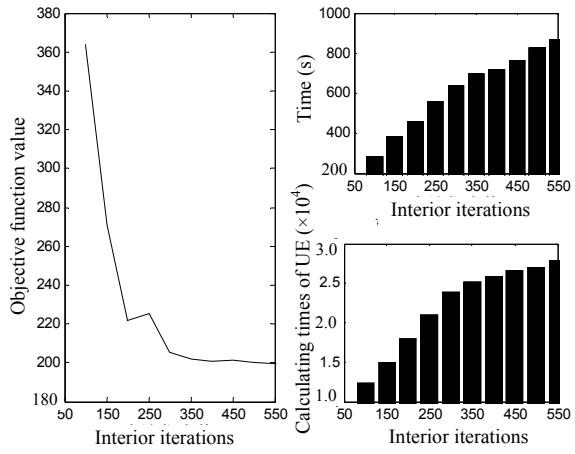

Fig. 3 Results in different interior iterations

5 Numeriacal example

The example used in this study is a network which consists of six nodes, 16 links and two OD pairs. The network is shown in Fig. 1 and the link cost function and demands can be referred to Suwansirikul *et al*.[14].

From the reaction between upper and lower level of the bilevel programming, the convergence level of the lower level models have a direct effects on the solution of the upper level models. However, the higher convergence level needs more time. We check the implementation time and different convergence levels, with the stop condition ε is set as 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}. It has been found that when $\varepsilon \leq 10^{-1}$, the optimal solution of the example is fluctuation on 201.5. With the increase of the convergence level, the implementation time of time(s) is also increasing. Combing time and convergence level, we set ε=0.01 in the following implementation.

5.1 OATD Method

We first set the benchmarking parameters combination of simulated annealing algorithm is with the inner iteration number M=350, the initial temperature T_0=10 000, the temperature decrease rate α=0.8, the termination temperature ε=0.001. Set the initial stepsize l_0=5, the minimize stepsize l_f=0.1, the field of decision variable y is y_a=0 and $\bar{y}_a = 10$. We consider the variation of these parameters (inner iteration number M, initial temperature T_0, temperature decrease rate α, termination temperature ε) in the following.

Fix the inner iteration number M, the temperature decrease rate α, the termination temperature ε, and let the initial temperature T_0 change in the field of $T_0 \in [1, 100\,000]$. The optimal solution and the implementation time are shown in Fig. 2. It can be found from Fig 2 that when the initial temperature T_0 is more than 1 000, the algorithm can converge to the optimal value 199.8. When the initial temperature T_0 is less than 1 000, the algorithm is easy to move the local optimal value. The objective value is moving between 200 and 245. Therefore, we can choose the initial temperature $T_0 \in [1\,000, 100\,000]$.

Fix the initial temperature T_0, the temperature decrease rate α, the termination temperature ε, and let the inner iteration number M be improved from 100 to 550, the optimal solution and the implementation time are shown in Fig. 3. It can be found that when the inner iteration number is small (M<350), the algorithm is easy to move to local optimal solution. When the inner number is larger than or equal to 350, the algorithm can converge to the optimal solution 199.6. From Fig. 3, it can be found that with the improvement of the inner iteration number, the implementation time increases to be linear. For the balance between the solution and the implementation time, the inner iteration can be chosen in the field of [350, 450].

Next, fix the inner iteration number M, initial temperature T_0, termination temperature ε, and set the temperature decrease rate α from 0.45 to 0.9. The optimal solution and implementation time are shown in Fig. 4. It can be found that the algorithm is easy to move to the local optimal value if the temperature decrease rate α<0.8. When the value $\alpha \geq 0.8$, the optimal solution is convergence to the value of 201.6. Generally, the bigger the value of α is, the larger the search space is. The algorithm can arrive at the optimal solution if it is set no less than 0.9. On the other hand, from Fig. 4, it can be found that the larger the value of α is, the slower the temperature decreasing is, and the longer the implementation time is. To keep the balance of precision and implementation time, the temperature decrease rate can be set in the field of [0.8, 0.9].

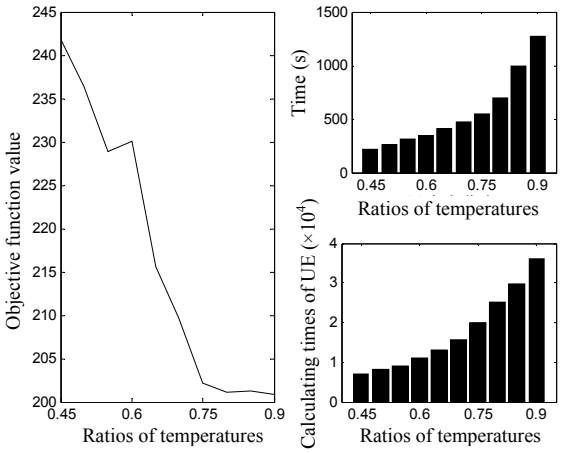

Fig. 4 Results in different ratios of temperatures

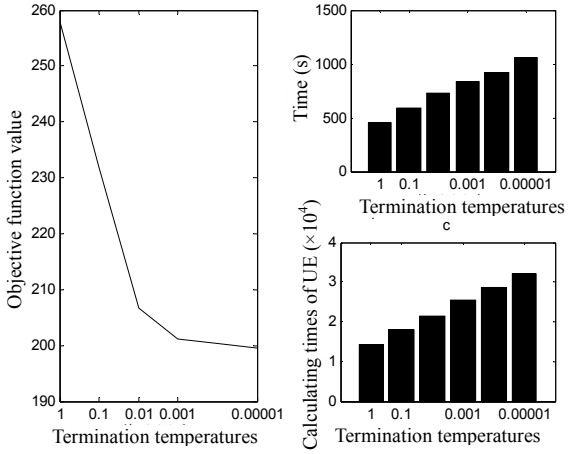

Fig. 5 Results in different termination temperatures

At last, fix the inner iteration number M, initial temperature T_0, temperature decrease rate α, and decrease the termination temperature ε from 1 to 0.0001. The implementing details are demonstrated in Fig. 5. Generally, to guarantee the optimal solution of the algorithm, the termination temperature ε needs to be set small enough. From Fig. 5, it can be found that the less the termination temperature ε is, the more the solution approaches to the optimal solution. When $\varepsilon \leqslant 0.001$, the solution approaches to 200.8. However, the iteration number will increase (the implementation time will add) with the small termination temperature ε. From Figs. 5, it can be found that the improving of the algorithm time is linear, at the rate about 200 seconds.

5.2 FD Method

According to the above discussion, the simulated annealing algorithm was separated into two cases to analyze, the best and worst case (relative best and relative worst). Under different cases, the two parameter levels (+ and –) are separated. The fields of parameters are based on the above OATD analysis. See Table 1 for details.

Table 1 Parameters levels

Case	Parameter level	M	ε	α
The best	+	[450,550]	[0.0001,0.00001]	[0.85,0.90]
	–	[350,450]	[0.001,0.0001]	[0.80,0.85]
The worst	+	[250,350]	[0.01,0.001]	[0.65,0.80]
	–	[150,250]	[1,0.01]	[0.50,0.65]

Table 2 Index effects on numerical results

Run	Parameters level						B	W
	M	ε	α	(M, ε)	(M, α)	(ε, α)	y	y
1	–	–	–	+	+	+	202.70	254.57
2	+	–	–	–	–	+	201.45	263.48
3	–	+	–	–	+	–	201.05	262.31
4	+	+	–	+	–	–	200.72	223.81
5	–	–	+	+	–	–	201.08	279.83
6	+	–	+	–	+	–	199.62	204.27
7	–	+	+	–	–	+	200.53	232.43
8	+	+	+	+	+	+	199.94	236.76
	–0.764	–0.549	–1	0.376	–0.098	0.452	B	Index
	–1.000	–0.464	–0.504	0.322	–0.413	0.168	W	

Note: B—the best; W—the worst.

For each case of the example, M, $\varepsilon$and α are chosen randomly from their two parameter levels, and there are six values in total. Under each parameter combination, it can be found that the solution and implementation time. Here we take about 10 times for the average value under each parameter combination. Furthermore, the initial temperature T_0 is set as fixed parameter as the field of T_0 is very small compared with the whole value field.

From Table 2, it can be found that the effects of the inner iteration and temperature decrease rate are clear, whose choice will directly affect the optimal solution of the algorithm. In different cases, the effects are different with the two parameters. In the best case, the temperature decrease rate makes greater influences, since the inner iteration number can guarantee that the algorithm can converge to the field of optimal solution. In the worst case, the inner iteration number makes great influences, as the searching process cannot approach the optimal solution under this case. It is easy to move to the local optimal solution with the adjustment of temperature decrease rate. Furthermore, the combination effect of the inner iteration number and temperature decrease rate is also very clear from Table 2, and the feasible solutions approach the optimal solution when the parameters are with the "+" level.

From Table 3, it can be found that both inner iteration number and temperature decrease rate affect the implementing time of the algorithm, and the effect of temperature decrease rate is larger. When the inner iteration number changes

Table 3 Index effects on implementing time

Run	Parameters level						B	W
	M	ε	α	(M, ε)	(M, α)	(ε, α)	Time	Time
1	–	–	–	+	+	+	589.3	33.656
2	+	–	–	–	–	+	866.2	90.625
3	–	+	–	–	+	–	696.6	53.844
4	+	+	–	+	–	–	1036.8	102.16
5	–	–	+	+	–	–	1253.3	98.625
6	+	–	+	–	+	–	1736.5	209.31
7	–	+	+	–	–	+	1470.5	95.063
8	+	+	+	+	+	+	2145.2	229.36
	0.519	0.264	1.000	0.074	0.158	0.102	B	Index
	0.994	0.136	1.000	0.042	0.396	–0.043	W	

Note: B—the best; W—the worst.

linearly, the implementing time of the algorithm improves linearly; however, when the temperature decrease rate changes linearly, the implementing time of the algorithm changes exponentially. It can be found that the implementing time of the algorithm is shorter with the parameters set "–" level.

6 Conclusions

In this paper, the authors used the simulated annealing algorithm to solve the CNDP, and demonstrated the feasibility with examples. The numerical example demonstrated the setting of parameters has a clear effect on the solution and implementing time. From OATD method, temperature decrease rate should be chosen rightly to guarantee the searching space, and inner iteration number is considered carefully. The termination temperature should be chosen small. It is set 0.001 in this example and the temperature decrease rate is set in [0.75 0.90]. From the FD method, the inner iteration number and the temperature decrease rate have clear effects on the simulated annealing algorithm. The choice of these two parameters is directly associated with the acquirement of the optimal solution. Under different status, the effect level is different with the two parameters: when the parameter is on the best case, the effect of the temperature decrease rate is larger; however, when the parameter is on the worst case, the effect of the inner iteration number is larger. Furthermore, the combination effects of the inner temperature number and temperature decrease rate are also clear.

Acknowledgements

This research was funded by the National Natural Science Foundation of China (Nos. 70771005, 70631001), and the programs for National Basic Research Program of China (973 Program, No. 2006CB705503).

References

[1] Gao Z Y, Song Y F, Si B F. Urban Traffic Network Design Problem: Theory and Method, Beijing: China Railway Publishing House, 2000.

[2] Abdulaal M, LeBlanc L J. Continuous equilibrium network design models. Transportation Research Part B, 1979, 13(1): 19–32.

[3] Boyce D E. Urban transportation network equilibrium and design models: recent achievements and future prospectives. Environment and Planning A, 1984, 16(11): 1445–1474.

[4] Yang H, Bell M G H. Models and algorithms for road network design: a review and some new developments. Transport Review, 1998, 18(3): 257–278.

[5] Liu C Q. Modern Urban Planning, Beijing: China Communications Press, 2001.

[6] Zhang G Q, Lu J. Genetic algorithm for continuous network design problem. Journal of Transportation Systems Engineering and Information Technology, 2007, 7(1): 101–105.

[7] Friesz T L, Cho H J, Mehta N J, *et al*. A simulated annealing approach to network design problem with variational inequality constraints. Transportation Science, 1992, 26(1): 17–26.

[8] Friesz T L, Tobin R, Shah S, *et al*. The multi-objective equilibrium network design problem revisited: a simulated annealing approach. European Journal of Operations Research, 1993, 65(1): 44–57.

[9] Gui L. Optimization models and algorithms of traffic network design. System Engineering, 2006, 24(12): 26–32.

[10] Liu C Q. Studies of models and algorithms of traffic network design problems. Journal of Highway and Transportation Research and Development, 2003, 20(12): 57–67.

[11] Xu T Z, Wei H, Wang Z D. Study on continuous network design problem using simulated annealing and genetic algorithm. Expert Systems with Applications, 2009, 36(2): 2735–2741.

[12] Saltelli A, Chan K, Scott E M. Sensitivity Analysis: Probability and Statistics Series, New York, USA: John Wiley & Sons Ltd., 2000.

[13] Chen A, Lee D H, Jayakrishnan R. Computational study of state-of-the-art path-based traffic assignment algorithms. Mathematics and Computers in Simulation, 2002, 59(6): 509–518.

[14] Suwansirikul C, Friesz T L, Tobin R L. Equilibrium decomposed optimization: a heuristic for the continuous equilibrium network design problem. Transportation Science, 1987, 21(4): 254–263.