World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:9, 2009

Transportation Under the Threat of Influenza

Yujun Zheng,Qin Song,Haihe Shi,and Jinyun Xue

Abstract—There are a number of different cars for transferring hundreds of close contacts of swine influenza patients to hospital, and we need to carefully assign the passengers to those cars in order to minimize the risk of influenza spreading during transportation. The paper presents an approach to straightforward obtain the optimal solution of the relaxed problems, and develops two iterative improvement algorithms to effectively tackle the general problem.

Keywords—Influenza spread; discrete optimization; stationary point; iterative improvement.

I. Introduction

SEVERAL students were found to have probable human swine influenza, and the school must transfer hundreds of close contacts to hospital for observation. There are a dozen cars of different capacities and with different sanitary and ventilation conditions, for each of which we assign a "threatening rate" th_i and evaluate the risk of influenza spreading as:

$$T_i(x_i) = th_i(b^{x_i} - 1) \tag{1}$$

where $b > 1$, $0 < th_i \leq 1$, and x_i is the number of passengers carried by car i. Suppose the maximum capacity of car i is c_i passengers, the number of cars is n and the total number of passengers is m, and thus the problem to minimize the risk of influenza spreading during transportation can be specified as follows:

$$\begin{aligned} \min \quad & f_n(m) = \sum_{i=1}^{n} th_i(b^{x_i} - 1) && (2)\\ s.t. \quad & \sum_{i=1}^{n} x_i = m && (3)\\ & x_i \in \mathbb{Z}_+, \quad 0 < i \leq n && (4)\\ & x_i \leq c_i, \quad 0 < i \leq n && (5) \end{aligned}$$

Without losing generality, we assume that $m < \sum_{i=1}^{n} c_i$. Under the same conditions, the car with less th_i should carry more passengers, and thus we can also preprocess the problem so that $\mathbf{th} = \{th_1, th_2, ..., th_n\}$ is sorted in increasing order. An obvious approach to the problem is to generate all subsets of integers [1], [2] that satisfy the capacity constraints and whose sum is m, and then using dynamic programming to obtain a solution with the minimum objective function value, which typically results in an algorithm that runs in $O(mn)$ time in the worst case [3].

In this paper we first present two relaxed versions of the problem and show that they can be solved in polynomial time, and then develops two algorithms that use iterative improvement to work out the optimal solution for the general problem, the effectiveness of which is demonstrated experimentally.

Y. Zheng, H. Shi, and J. Xue are with the Institute of Software and the Graduate University of the Chinese Academy of Sciences, Beijing 100080, PR China (e-mail: yujun.zheng@computer.org).

Q. Song is with the Department of Biotechnology, Beijing University of Agriculture.

II. The Problem without Capacity and Integer Constraints

First we only consider the relaxed problem with objective (2) and constraint (3), which is a nonlinear programming problem (NLP). Note for $n = 2$ differentiating yields:

$$\begin{aligned} & \frac{\partial f_2(m)}{\partial x_1} \\ = \quad & \frac{\partial (th_1(b^{x_1} - 1) + th_2(b^{m-x_1} - 1))}{\partial x_1} \\ = \quad & th_1 b^{x_1} \ln b - th_2 b^{m-x_1} \ln b \end{aligned}$$

At the stationary point $\frac{\partial f_2(m)}{\partial x_1} = 0$, the optimal solution can be directly obtained as:

$$x_1 = (m + \log_b \frac{th_2}{th_1})/2 \tag{6}$$

$$x_2 = (m + \log_b \frac{th_1}{th_2})/2 \tag{7}$$

$$f_2^*(m) = 2(b^m th_1 th_2)^{1/2} - th_1 - th_2 \tag{8}$$

Such a result is scalable and we have the following theorem:

Theorem 1. The optimal solution of the relaxed NLP is as follows:

$$x_i = (m + \log_b \frac{\prod_{i=1}^{n} th_i}{th_i^n})/n, \quad 0 < i \leq n \tag{9}$$

$$f_n^*(m) = n(b^m \prod_{i=1}^{n} th_i)^{1/n} - \sum_{i=1}^{n} th_i \tag{10}$$

Proof (Mathematical induction). The theorem is trivial for $n = 1$ and has been proved for $n = 2$. Next we assume Equation (9) holds for $n = k$ and thus have:

$$\begin{aligned} & f_k^*(m) \\ = \quad & \sum_{i=1}^{k} th_i(b^{x_i} - 1) \\ = \quad & \sum_{i=1}^{k} th_i(b^m \frac{\prod_{i=1}^{k} th_i}{th_i^k})^{1/k} - \sum_{i=1}^{k} th_i \\ = \quad & k(b^m \prod_{i=1}^{k} th_i)^{1/k} - \sum_{i=1}^{k} th_i \end{aligned}$$

Now for $n = k + 1$ it is obvious to conclude:

$$f_{k+1}(m) = \min \{f_k^*(m - x_{k+1}) + th_{k+1}(b^{x_{k+1}} - 1)\} \tag{11}$$

Therefore we have:

$$\begin{aligned} & \frac{\partial f_{k+1}(m)}{\partial x_{k+1}} \\ = \quad & \frac{\partial f_k^*(m - x_{k+1})}{\partial x_{k+1}} + \frac{\partial (th_{k+1}(b^{x_{k+1}} - 1))}{\partial x_{k+1}} \\ = \quad & (-1/k)k(b^{m-x_{k+1}} \prod_{i=1}^{k} th_i)^{1/k} \ln b + th_{k+1} b^{x_{k+1}} \ln b \\ = \quad & (th_{k+1} b^{x_{k+1}} - (b^{m-x_{k+1}} \prod_{i=1}^{k} th_i)^{1/k}) \ln b \end{aligned}$$

Let $\frac{\partial f_{k+1}(m)}{\partial x_{k+1}} = 0$ we get:

$$\begin{aligned}
& (th_{k+1}b^{x_{k+1}})^k = b^{m-x_{k+1}} \prod_{i=1}^{k} th_i \\
\Leftrightarrow \quad & \frac{(th_{k+1})^k}{\prod_{i=1}^{k} th_i} = b^{(m-(k+1)x_{k+1})} \\
\Leftrightarrow \quad & \log_b \frac{(th_{k+1})^k}{\prod_{i=1}^{k} th_i} = m-(k+1)x_{k+1} \\
\Leftrightarrow \quad & x_{k+1} = (m - \log_b \frac{(th_{k+1})^k}{\prod_{i=1}^{k} th_i})/(k+1) \\
\Leftrightarrow \quad & x_{k+1} = (m + \log_b \frac{\prod_{i=1}^{k+1} th_i}{(th_{k+1})^{k+1}})/(k+1)
\end{aligned}$$

And the new optimal objective value is:

$$\begin{aligned}
& f^*_{k+1}(m) \\
= \quad & f^*_k(m-x_{k+1}) + th_{k+1}(b^{x_{k+1}}-1) \\
= \quad & k(b^{m-x_{k+1}} \prod_{i=1}^{k} th_i)^{1/k} \\
& \quad - \sum_{i=1}^{k} th_i + th_{k+1}b^{x_{k+1}} - th_{k+1} \\
= \quad & k(\frac{b^m th_{k+1} \prod_{i=1}^{k} th_i}{(b^m \prod_{i=1}^{k+1} th_i)^{k+1}})^{1/k} \\
& \quad + th_{k+1} \frac{(b^m \prod_{i=1}^{k+1} th_i)^{1/(k+1)}}{th_{k+1}} - \sum_{i=1}^{k+1} th_i \\
= \quad & k(b^m \prod_{i=1}^{k+1} th_i)^{1/(k+1)} \\
& \quad + (b^m \prod_{i=1}^{k+1} th_i)^{1/(k+1)} - \sum_{i=1}^{k+1} th_i \\
= \quad & (k+1)(b^m \prod_{i=1}^{k+1} th_i)^{1/(k+1)} - \sum_{i=1}^{k+1} th_i
\end{aligned}$$

Thus the theorem is proven, which implies that the relaxed problem can be directly solved in $O(n)$ time (given that logarithms and exponentials can be computed efficiently [4]). □

III. The Problem without Capacity Constraints

Next we consider the relaxed problem with objective (2) and constraint (3) and (4), which is a nonlinear integer programming problem (NLIP). A general approach to the problem consists of the following two stages:

1) Develop an initial solution;
2) Continually improve the solution by moving a passenger from one car to another until the objective function cannot be reduced further.

Suppose there are x_i passengers in car i ($x_i \geq 1$) and x_j passengers in car j, moving one passenger from i to j will result a change of the objective function as follows:

$$\begin{aligned}
& \Delta_{i,j} \qquad (12) \\
= \quad & th_i(b^{x_i-1}-1) + th_j(b^{x_j+1}-1) \\
& \quad - th_i(b^{x_i}-1) - th_j(b^{x_j}-1) \\
= \quad & (b-1)(th_j b^{x_j} - th_i b^{x_i-1}) \qquad (13)
\end{aligned}$$

To ensure $\Delta_{i,j} < 0$ we must have $th_j b^{x_j} < th_i b^{x_i-1}$, or $x_i - x_j > 1 + \log_b \frac{th_j}{th_i}$. On the other hand, the difference between moving a passenger from i to j and that from i to k is as follows:

$$\begin{aligned}
& \Delta_{i,j} - \Delta_{i,k} \\
= \quad & (b-1)(th_j b^{x_j} - th_i b^{x_i-1}) - (b-1)(th_k b^{x_k} - th_i b^{x_i-1}) \\
= \quad & (b-1)(th_j b^{x_j} - th_k b^{x_k})
\end{aligned}$$

which suggests the car with less $th_j b^{x_j}$ should be preferably chosen as the moving target.

To minimize the computational effort, we should start with a solution as close to the optimum as possible, and the effective result of above section provides a good start. The following lemma demonstrates that the optimum of the NLIP is close to that of the NLP:

Lemma 1. Let $\mathbf{x} = \{x_1, x_2, ...x_n\}$ be the the optimal solution of the relaxed NLIP and $\mathbf{x^0} = \{x_1^0, x_2^0, ...x_n^0\}$ be that of the relaxed NLP, then for any i ($0 < i \leq n$) we have:

$$\lfloor x_i^0 \rfloor \leq x_i \leq \lceil x_i^0 \rceil \qquad (14)$$

Proof. By contradiction we first assume there is an i such that $x_i > x_i^0 + 1$, then there must be at least a $j \neq i$ satisfying $x_j < x_j^0$. Therefore we have:

$$\begin{aligned}
& th_j b^{x_j} - th_i b^{x_i-1} \\
< \quad & th_j b^{x_j^0} - th_i b^{x_i^0} \\
= \quad & th_j (\frac{b^m \prod_{k=1}^{n} th_k}{(th_j)^n})^{1/n} - th_i (\frac{b^m \prod_{k=1}^{n} th_k}{(th_i)^n})^{1/n} \\
= \quad & 0
\end{aligned}$$

Thus we can always move a passenger from i to j to reduce the objective function, which says $\mathbf{x}$ is not the optimal solution of the problem.

Similarly, if there is an i such that $x_i < x_i^0 - 1$, then there must be at least a $j \neq i$ satisfying $x_j > x_j^0$ and we can get the same contradiction. □

Now we propose an efficient algorithm for the relaxed NLIP, which starts with an initial solution in which $x_i = \lfloor x_i^0 \rfloor$ ($2 \leq i \leq n$) and $x_1 = m - \sum_{i=2}^{n} x_i$, and continually moves one passenger from x_1 to an x_i whose $th_i b^{x_i}$ is minimum and $\Delta_{1,i} < 0$, as illustrated in Algorithm 1. Including the time required for sorting [5], the running time of Algorithm 1 is $O(n \log n)$.

Before going deep into the general problem with capacity constraints, we should point out that, in many practical cases, the solution produced by Algorithm 1 actually satisfies the capacity constraints, given that there is typically no big difference between the cars' capacities and there is always a certain surplus of the total capacity.

IV. The General Constrained Problem

In this section we develop two algorithms for the general version of the problem, both using the iterative improvement approach based on heuristics indicated by Equation (12). The first starts with a feasible solution and the second starts with a probably infeasible solution.

Min-Threat-Trans$(m, n, \mathbf{th}, b)$

begin
  $ps \leftarrow \prod_{i=1}^{n} \mathbf{th}[i]$;
  $\mathbf{x}[1] \leftarrow m$;
  for $i \leftarrow 2$ to n **do** // initial solution
    $\mathbf{x}[i] \leftarrow \lfloor (m + \log_b \frac{ps}{(\mathbf{th}[i])^n})/n \rfloor$;
    IN-SORT$(\mathbf{A}, \{\mathbf{x}[i], \mathbf{th}[i]\})$;
    $\mathbf{x}[1] \leftarrow \mathbf{x}[1] - \mathbf{x}[i]$;
  endfor;
  $\mathbf{A}$.InsertAt$(0, \{\mathbf{x}[1], \mathbf{th}[1]\})$;
  for $i \leftarrow 2$ to n **do** // iterative improvement
    if $(\mathbf{A}[i] < \mathbf{A}[1]/b)$ **then begin**
      $\mathbf{A}[1].x \leftarrow \mathbf{A}[1].x - 1$;
      $\mathbf{A}[i].x \leftarrow \mathbf{A}[i].x + 1$;
    end
    else return;
  endfor;
end

Algorithm 1: The iterative improvement algorithm for the minimum flu-spread risk problem without capacity constraints, where the procedure IN-SORT inserts an element $\{x, th\}$ into list $\mathbf{A}$ and keeps the list sorted in increasing order of $\mathbf{x}[i] \cdot b^{\mathbf{th}[i]}$ $(0 < i \leq |\mathbf{A}|)$.

A. Starting with a Feasible Solution

The first general algorithm tries to fill the cars with small threatening rate to get an initial solution, and then iteratively moves a passenger from car i with maximum $th_i b^{x_i}$ to car j with minimum $th_i b^{x_i}$ to yield the largest decrease of the objective function, as suggested by Equation (12).

Based on same technique used in the above section, the algorithm maintains $th_i b^{x_i}$ $(0 < i \leq n)$ in increasing order in list $\mathbf{A}$, and continually moves a passenger from the last element to the first element of $\mathbf{A}$ until the objective function cannot be reduced further, as illustrated in Algorithm 2.

Suppose that during the first stage k cars have been filled, then the second loop of Algorithm 2 will execute $r = \sum_{i=1}^{k}(c_i - x_i)$ times, and hence the total running time of Algorithm 2 is $O(n \log n + nr)$.

B. Starting with an Infeasible Solution

The second general algorithm starts with an initial solution close to the NLP's optimal solution described in Section II. If the initial solution is feasible, we can employ Algorithm 1 to produce the exact optimal solution for the problem, otherwise we iteratively reduce the violation of constraints until the solution becomes feasible. In detail, the cars are grouped into two lists $\mathbf{A}$ and $\mathbf{B}$, where $\mathbf{A}$ contains the cars whose x_i exceeds c_i and $\mathbf{B}$ contains the cars with $x_i < c_i$; while $\mathbf{A}$ is not empty, we move a passenger from the maximum element of $\mathbf{A}$ to the minimum element of $\mathbf{B}$, as illustrated in Algorithm 3.

Let k denotes the initial size of $\mathbf{A}$, then the second loop of Algorithm 3 will execute $r' = \sum_{i=1}^{k}(x_i - \lfloor x_i^0 \rfloor)$ times, and the total running time of the algorithm is $O(n \log n + nr')$.

Min-Threat-Trans-Dec$(m, n, \mathbf{th}, \mathbf{c}, b)$

begin
  $m' \leftarrow m$; $i \leftarrow 1$;
  while $m' > 0$ **do** // initial solution
    if $(\mathbf{c}[i] < m')$ **then begin**
      $\mathbf{x}[i] \leftarrow \mathbf{c}[i]$;
      $m' \leftarrow m' - \mathbf{c}[i]$;
    end
    else $\mathbf{x}[i] \leftarrow m'$;
    IN-SORT$(\mathbf{A}, \{\mathbf{x}[i], \mathbf{th}[i]\})$;
    $i \leftarrow i + 1$;
  endwhile;
  for $j \leftarrow i$ to n **do** $\mathbf{A}$.InsertAt$(0, \{0, \mathbf{th}[j]\})$;
  while $\mathbf{A}[1] < \mathbf{A}[n]/b$ **do** // iterative improvement
    $\mathbf{A}[1].x \leftarrow \mathbf{A}[1].x + 1$;
    $\mathbf{A}[n].x \leftarrow \mathbf{A}[n].x - 1$;
    IN-SORT$(\mathbf{A}, \mathbf{A}[1])$;
    IN-SORT$(\mathbf{A}, \mathbf{A}[n])$;
  endwhile;
end

Algorithm 2: The minimum flu-spread risk algorithm that iteratively decreases the objective function.

Min-Threat-Trans-Inc$(m, n, \mathbf{th}, \mathbf{c}, b)$

begin
  $ps \leftarrow \prod_{i=1}^{n} \mathbf{th}[i]$; $\mathbf{x}[1] \leftarrow m$;
  for $i \leftarrow 2$ to n **do** // initial solution
    $\mathbf{x}[i] \leftarrow \lfloor (m + \log_b \frac{ps}{(\mathbf{th}[i])^n})/n \rfloor$;
    if $\mathbf{x}[i] > \mathbf{c}[i]$ **then** IN-SORT$(\mathbf{A}, \{\mathbf{x}[i], \mathbf{th}[i]\})$;
    else if $\mathbf{x}[i] < \mathbf{c}[i]$ **then** IN-SORT$(\mathbf{B}, \{\mathbf{x}[i], \mathbf{th}[i]\})$;
    $\mathbf{x}[1] \leftarrow \mathbf{x}[1] - \mathbf{x}[i]$;
  endfor;
  if $\mathbf{x}[1] > \mathbf{c}[1]$ **then** IN-SORT$(\mathbf{A}, \{\mathbf{x}[1], \mathbf{th}[1]\})$;
  else if $\mathbf{x}[1] < \mathbf{c}[1]$ **then** IN-SORT$(\mathbf{B}, \{\mathbf{x}[1], \mathbf{th}[1]\})$;
  while $|A| > 0$ **do** // iterative improvement
    $\mathbf{A}[n].x \leftarrow \mathbf{A}[n].x - 1$;
    if $\mathbf{A}[n].x = \mathbf{A}[n].c$ **then** $\mathbf{A}$.RemoveAt(n);
    else IN-SORT$(\mathbf{A}, \mathbf{A}[n])$;
    $\mathbf{B}[1].x \leftarrow \mathbf{B}[1].x + 1$;
    if $\mathbf{B}[1].x = \mathbf{B}[1].c$ **then** $\mathbf{B}$.RemoveAt(1);
    else IN-SORT$(\mathbf{B}, \mathbf{B}[1])$;
  endwhile;
end

Algorithm 3: The minimum flu-spread risk algorithm that iteratively reduce the violation of constraints and increases the objective function.

TABLE I
THE RANGES OF PROBLEM VARIABLES USED IN THE TEST CASES

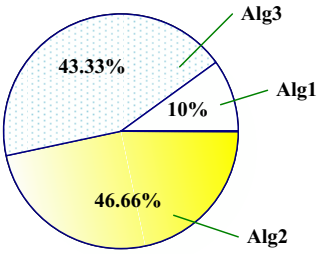
Variable	min	max	average
b	1.05	1.55	1.3
m	100	1500	800
n	5	20	10
th_i	0.1	1	0.55
c_i	10	100	25

C. Experimental Comparison of the Algorithms

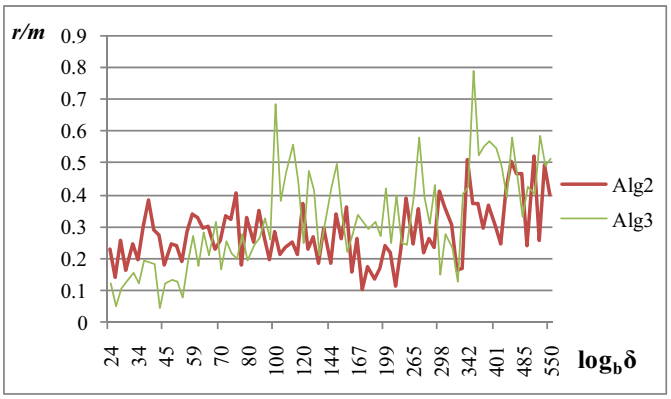
Roughly speaking, the closer the optimum of the general problem is to the optimum of the relaxed NLP, the more efficient Algorithm 3 will be than Algorithm 2. However, it is difficult to accurately estimate the algorithm efficiency before we reach the optimal solution. In order to evaluate the algorithm efficiency and find some guidelines for algorithm selection, we randomly generate 90 problem instances and collect the iteration times of the algorithms. Table 1 presents the ranges of variables used in these test cases.

The experimental result shows that there are 9 cases (which are deemed to be reasonably practical) can be directly solved by Algorithm 1; among the remaining 81 cases, there are 39 cases on which the Algorithm 3 is more efficient than Algorithm 2, as illustrated in Fig 1 (a). Let r_2 be the iteration times of Algorithm 2 and r_3 be that of Algorithm 3, we get that the average values of r_2/m and r_3/m are 28.2% and 30.1% respectively, which demonstrates that both the algorithms overwhelm the basic dynamic programming approach; moreover, the average value of $\min(r_2, r_3)/m$ is 23.3%, which states that the appropriate algorithm selection will lead to a significant performance improvement.

Since the iteration times of the algorithms are highly depended on the difference of $th_i b^{x_i}$ between different cars, here we compute $p = (\sum_{i=1}^{n} th_i b^{m/n})/n$ and $\delta = \sum_{i=1}^{n}(th_i b^{c_i} - p)^2/n$ for each problem instance, and evaluate the algorithm performance over the instances based on δ. Fig 1 (b) illustrates the changes of algorithm performance (evaluated by r/m) with respect to $\log_b \delta$, from which we can see that Algorithm 3 is typically efficient than Algorithm 2 if $\log_b \delta < 95$, which gives an useful criterion for selecting the appropriate algorithm.



(a)



(b)

Fig. 1. Experimental comparison of the algorithms.

V. CONCLUSION AND DISCUSSION

The paper presents the problem of minimizing the risk of influenza spreading during transportation and develops an efficient algorithmic approach to the problem. This problem comes from our recent work in contingency planning for influenza epidemics and is specified based on the dynamic models of epidemic spreading [6], [7]. For other epidemic diseases (such as bird influenza, malaria, encephalitis, etc), we can adjust the value of the constant b to match their spreading models as well as possible. Furthermore, the formulation can also be modified or extended to represent problems in other domains such as freight transportation (e.g., [8], [9]), vehicle routing [10], resource allocation, and network design. For example, suppose a large amount of materiel needs to be transported to the battlefield through several paths with different threatening rate, and the risk of being attacked increases exponentially with the amount of materiel; Such a military transportation problem can be formulated exactly the same way and be solved by the same algorithms proposed in this paper.

According to the actual demands of epidemic prevention, we are extending our problem model by a) considering the condition that the total capacity is not enough, and thus additional rounds of transportation are needed; b) adding more realistic constraints, e.g., some passengers must be in the same car; c) including more objectives, e.g., to minimize transportation time and costs, and thus transform the problem into a multi-objective programming problem. Ongoing efforts also include developing polynomial-time approximate algorithms for the problem and its extensions.

ACKNOWLEDGMENT

The work was supported in part by grants from National Natural Science Foundation (No. 60773054) of China.

REFERENCES

[1] S.G. Akl, "A comparison of combination generation methods", *ACM Tran. Math. Soft.*, vol. 7, 1981, pp. 42–45.
[2] D.R. Baronaigien, F. Ruskey, "Efficient generation of subsets with a given sum", *J. Combin. Math. Combin. Comput.*, vol. 14, 1993, pp. 87–96.
[3] S. Martello, P. Toth, "A mixture of dynamic programming and branch-and-bound for the subset-sum problem", *Management Sci.*, vol. 30, 1984, pp. 765–771.
[4] K. McCurley, "The discrete logarithm problem, In: Cryptology and Computational Number Theory", *Proceedings of Symposia in Applied Mathematics*, vol. 42, 1990, pp. 49–74.
[5] M. Ben-Or, "Lower bounds for algebraic computation trees", In: *15th Annual ACM Symposium on theory of Computing*, ACM Press, New York, 1983, pp. 80–86.

[6] N. Noah, M. O'Mahony, *Communicable Disease - Epidemiology and Control*, John Wiley, New York, 1998.
[7] Z. Ma, J. L, "Basic Knowledge and Developing Tendencies in Epidemic Dynamics", *Mathematics for Life Science and Medicine*, Springer Berlin-Heidelberg, 2007, pp. 5–49.
[8] T.G. Crainic, J.M. Rousseau, "Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem", *Transportation Research Part B: Methodological*, vol. 20, 1986, pp. 225–242.
[9] M. Zhang, G.S. Liu, L.K. Wu, Y.H. He, "Model and algorithm for bilevel transportation problem", *Acta Mathematicae Applicatae Sinica*, English Series, vol. 31, 2008, pp. 17–23.
[10] T.K. Ralphs, L. Kopman, W.R. Pulleyblank, L.E. Trotter, "On the capacitated vehicle routing problem", *Math. Prog.*, vol. 94, 2003, pp. 343–359.