Available online www.jocpr.com

Journal of Chemical and Pharmaceutical Research, 2014, 6(5):2013-2018

Research Article

ISSN : 0975-7384
CODEN(USA) : JCPRC5

A combination algorithm for selecting functional logistics service vendors based on SQP and BNB

Meiling He[1*], Qifan Hu[2], Xiaohui Wu[1] and Peng Jing[1]

[1]*School of Automotive and Traffic Engineering, Jiangsu University, Zhenjiang, China*
[2]*College of Business Administration, Nanchang Institute of Technology, Nanchang, China*

ABSTRACT

When logistics services integrator integrates functional logistics service vendors, the determination of the number is critical. Considering the reliability of vendors, the paper formulates an integer non-linear programming model to solve the vendor selection problem. The objective function is to minimize the total cost. The constraint functions are to maximize the reliability and choose the appropriate vendors. The reliability of logistics service supply chain is studied. This paper constructs a combination algorithm based on sequential quadratic programming and branch and bound method. It can meet both non-linear programming and integer programming. By numerical simulation, the optimal solution and near optimal solution are given. The results show the suitable numbers of each kind of functional logistics service vendors, the minimum cost and the maximum reliability of logistics services supply chain.

Keywords: Combination algorithm, sequential quadratic programming, branch and bound, integer non-linear programming, functional logistics service vendors

---

INTRODUCTION

In the process of building the logistics service supply chain, logistics services integrator is predominant. To ensure high reliability of logistics service supply chain, the integrator integrates multiple functional logistics service vendors. But increase the quantity blindly will bring great cost. Therefore, how to reduce the cost of seeking logistics service vendors is the key to the logistics service supply chain operation. At present, techniques about vendors selection are focused on the linear weighting method [1-4], cost method [5, 6], mathematical programming method [7-9] and combination method [10, 11], etc. Generally speaking, the nonlinear programming problem is much more difficult than the linear. It requires the variable must be an integer. Many existing methods are restricted. There is no general algorithm suitable for the problem and each method has its own specific scope [12]. Literature [13] presents a non-zero integer non-linear goal programming model that minimizes the number of maintenance workforce while maximizing their productivity. The branch and bound technique is more successful computationally than the cutting-plane technique or the implicit enumeration technique [14]. This paper proposes a combination algorithm based on improved sequential quadratic programming and branch and bound method.

2. Model description

The subject of this study is functional logistics service vendors which provide logistics services for logistics services integrator. They are integrated by logistics services integrator in building logistics network. The functional logistics service vendors mainly include transportation service vendors (TSV), storage service vendors (SSV), distribution service vendors (DSV) and other service vendors (OSV). The minimum reliability of logistics service supply chain is given. Those notations are used for the proposed model: x_i is the number of the chosen functional logistics service vendors of category *i*, $i \in [1,2,...n]$. $R(0)$ is the minimum reliability of logistics service supply chain. R

is the reliability of logistics service supply chain. R_i is the reliability of functional logistics service vendors of category *i*. c_i is the cost of functional logistics service vendors of category *i*.

The objective function is cost function $\sum_{i=1}^{n} c_i \cdot x_i$. The reliability R must be greater than the minimum. All kinds of logistics service vendors must be chosen at least one.

$$\text{Min } Z = \sum_{i=1}^{n} c_i \cdot x_i \tag{1}$$

$$\text{s.t.} \quad R \geq R(0) \tag{2}$$

$$R = \prod_{i=1}^{n} (1 - (1 - R_i)^{x_i}) \tag{3}$$

$$x_i \geq 1, \ \ x_i \text{ is an integer, } i \in [1,2,...n] \ . \tag{4}$$

The model is an integer non-linear programming problem. Murty proved that the non-linear programming problem is NP-hard problem [15]. There is no equality constraint in the original problem. It can be summarized as the following:

$$\text{Min } f(x) \tag{5}$$

$$\text{s.t.} \ \ g_i(x) \leq 0, \quad i \in I \ . \tag{6}$$

3. Combination algorithm

Sequential quadratic programming (SQP). Define active set as $I(x) = \{i \in I : g_i(x) = 0\}$. If constrained index set is known as $A(x) = J \cup I(x)$, J is the index set of equality constraint [16]. The approximate actively set is $\mathbf{A}(\mathbf{x};\varepsilon) = \{i : g_i(\mathbf{x}) + \varepsilon\rho(\mathbf{x}, \lambda(\mathbf{x})) \geq 0\}$, here ε is a non-negative parameter, $\rho(\boldsymbol{x}, \lambda(\boldsymbol{x}) = \sqrt{\|\Phi(\boldsymbol{x}, \lambda)\|}$,

$$\Phi(\boldsymbol{x}, \boldsymbol{\lambda}(\boldsymbol{x})) = \begin{bmatrix} \nabla_x L(\boldsymbol{x}, \lambda(\boldsymbol{x})) \\ \min\{-f(\boldsymbol{x}), \lambda(\boldsymbol{x})\} \end{bmatrix} \quad , \quad f(\boldsymbol{x}) = \begin{bmatrix} g_1(\boldsymbol{x}) \\ g_2(\boldsymbol{x}) \\ \vdots \\ g_m(\boldsymbol{x}) \end{bmatrix} \quad ,$$

$$\lambda(\boldsymbol{x}) = -(\nabla \boldsymbol{g}(\boldsymbol{x})^T \nabla \boldsymbol{g}(\boldsymbol{x}) + diag(g_i(\boldsymbol{x}))^2)^{-1} \nabla \boldsymbol{g}(\boldsymbol{x})^T \nabla f(\boldsymbol{x}).$$

Then optimization problem with inequality constraints can be turned into pure equality constraint optimization problem.

$$\text{Min } f(\boldsymbol{x}) \tag{7}$$

$$\text{s.t.} \ \ \boldsymbol{g}(\boldsymbol{x}) = 0 \tag{8}$$

Here, $f : R^n \to R, \quad \boldsymbol{g} : R^n \to R^J$ is continuously differentiable.

The minimum point of the above problem is optimum solution of Eqs. (5) and (6).

The Lagrange function of the above problem is $\boldsymbol{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$. The gradient matrix of constraint function $\boldsymbol{g}$ is $\nabla \boldsymbol{g}(\boldsymbol{x}) = (\nabla g_1(\boldsymbol{x}), \nabla g_2(\boldsymbol{x}), \cdots)$, here Jacobi matrix is $\boldsymbol{B}(\boldsymbol{x}) = \nabla \boldsymbol{g}(\boldsymbol{x})^T$.

For iteration point ($\boldsymbol{x}^k, \boldsymbol{\lambda}^k$), programming (7) and (8) are approximate to the following quadratic programming problem.

$$\underset{\mathrm{d}}{\mathrm{Min}} \quad \frac{1}{2} d^T H(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) d + \nabla f(\boldsymbol{x}^k)^T d \tag{9}$$

$$\mathrm{s.t.} \quad B(\boldsymbol{x}^k) d + \boldsymbol{g}(\boldsymbol{x}^k) = 0 \tag{10}$$

The Newton direction $\boldsymbol{p}^k = (\boldsymbol{p}_x^k, \boldsymbol{p}_\lambda^k)^T$ meets the following equation:

$$\boldsymbol{K}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) \boldsymbol{p}^k = -\nabla L(\boldsymbol{x}^k, \boldsymbol{\lambda}^k). \tag{11}$$

$$\boldsymbol{P}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) = (\boldsymbol{p}_x^k)^T (\boldsymbol{H}_k^2 + \boldsymbol{B}_k^T \boldsymbol{B}_k) \boldsymbol{p}_x^k - 2 \boldsymbol{p}_x^k \boldsymbol{H}_k \boldsymbol{B}_k^T \boldsymbol{p}_\lambda^k + (\boldsymbol{p}_\lambda^k)^T \boldsymbol{B}_k \boldsymbol{B}_k^T \boldsymbol{p}_\lambda^k \tag{12}$$

The Newton direction $\boldsymbol{p}^k = (\boldsymbol{p}_x^k, \boldsymbol{p}_\lambda^k)^T$ meets the following

$$(\boldsymbol{p}^k)^T \nabla P(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) = -2P(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) \le 0. \tag{13}$$

Therefore, $\boldsymbol{p}^k$ is a non increase direction of function $P(\boldsymbol{x}, \boldsymbol{\lambda})$ in the point $(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)$.

Summarize the specific algorithm process.

Step 1: Initialization Select the initial point $(\boldsymbol{x}^0, \boldsymbol{\lambda}^0)$, allowable error $\tau > 0$, $\beta \in (0,1)$, $\delta > 2$, $\eta > 1$, $\rho > 0$, $\gamma > 0$, $k = 0$. **Step 2: Test the convergence** Calculate $\nabla f^k = \nabla f(\boldsymbol{x}^k)$, $\boldsymbol{H}_k = \boldsymbol{H}(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)$, $\boldsymbol{g}^k = \boldsymbol{g}(\boldsymbol{x}^k)$ and $\boldsymbol{B}_k = \boldsymbol{B}(\boldsymbol{x}^k)$. If $P(\boldsymbol{x}^k, \boldsymbol{\lambda}^k) \le \tau$, the approximate KKT point $(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)$ of Eqs. (7) and (8) can be got. Then, the algorithm stops. Otherwise, turn to step 3. **Step 3: Calculate the main search direction** Calculate the descent direction By solving (9) and (10), we get $\boldsymbol{d}_0^k$ and $\boldsymbol{\lambda}^*$. Calculate the feasible direction

$$\begin{bmatrix} \boldsymbol{H}_k & \nabla \boldsymbol{g}_{A^k}(\boldsymbol{x}^k) \\ \nabla \boldsymbol{g}_{A^k}(\boldsymbol{x}^k)^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{d} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\nabla f(\boldsymbol{x}^k) \\ -\left\| \boldsymbol{d}_0^k \right\|^\delta e_{A^k} \end{bmatrix}. \tag{14}$$

Here $e_{A^k} = (1,...,1)^T$, we can get $\boldsymbol{d}_1^k$. Calculate the feasible descent direction $\boldsymbol{d}_*^k$.

$$\boldsymbol{d}_*^k = \boldsymbol{d}_0^k + \rho_k \boldsymbol{d}_1^k. \tag{15}$$

$$\rho_k = \begin{cases} \rho & \nabla f(\mathbf{x}^k)^T \mathbf{d}_1^k \le 0, \\ \dfrac{-\nabla f(\mathbf{x}^k)^T \mathbf{d}_0^k}{\nabla f(\mathbf{x}^k)^T \mathbf{d}_1^k + \gamma} & \textit{others.} \end{cases} \tag{16}$$

Here, $\boldsymbol{p}_x^k = \boldsymbol{d}_*^k$, $\boldsymbol{p}_\lambda^k = \boldsymbol{\lambda}^* - \boldsymbol{\lambda}^k$, $\alpha = 1$.**Step 4: Choose the step size** If $P(\boldsymbol{x}^k + \alpha \boldsymbol{p}_x^k, \boldsymbol{\lambda}^k + \alpha \boldsymbol{p}_\lambda^k) \le (1 - \beta\alpha) P(\boldsymbol{x}^k, \boldsymbol{\lambda}^k)$, $\alpha_k = \alpha$, turn to step 5. Otherwise, $\alpha / \eta \to \alpha$, turn to step4. **Step 5: Correction the iteration point** $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha_k \boldsymbol{p}_x^k$, $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \alpha_k \boldsymbol{p}_\lambda^k$, $k+1 \to k$, turn to step 2. End.

Branch and bound (BNB). Branch and bound is an effective method for solving integer programming problems [17]. $D \subset A \subset R^n$ is non-empty feasible region. *A* is an open set. $f: A \to R$. If $\Omega \subset R^n$, subject $\mathsf{M} = \{M_i \mid i \in I\}$ is a part of Ω. $\Omega = \bigcup_{i \in I} M_i$. For $\forall i, j \in I$, $i \neq j$, $M_i \cap M_j = \partial M_i \cap \partial M_j$. Here, ∂M_i is the relative boundary of M_i. The specific algorithm is:

Step 1: Initialization Define a relaxation set $M_0 \supset D$. For each $M_i, i \in I$, its objective function's lower bound in $M_i \cap D$ is $\beta(M_i)$ and upper bound is $\alpha(M_i)$. The feasible point set $S(M_0) \subset D$. $\mathsf{M}_0 = \{M_0\}$, $\beta_0 = \beta(M_0)$, $\alpha_0 = \min\{f(S(M_0)), +\infty\}$. If $\alpha_0 < +\infty$, $\boldsymbol{x}^0 \in \arg\min f(S(M_0))$, *k*=1. **Step 2: Test the convergence** For M_0 and $M \in \mathsf{M}_{k-1}$, the feasible point set $S(M) \subset M \cap D$. $\beta_{k-1} = \min_{M \in \mathsf{M}_{k-1}} \beta(M)$, $\alpha_{k-1} = \min_{M \in \mathsf{M}_{k-1}} \alpha(M)$. If $\alpha_{k-1} - \beta_{k-1} = 0$, the point $\boldsymbol{x}^{k-1} \in D$ fitted with $f(\boldsymbol{x}^{k-1}) = \alpha_{k-1}$ is the optimal solution of Eqs. (5) and (6), and the algorithm stops. Otherwise, turn to step 3. **Step 3: Branch** Define $Q_k = \{M \in \mathsf{M}_{k-1} \mid \beta(M) < \alpha_{k-1}\}$. Select a non-null family of set $T_k \subset Q_k$ and construct each subdivision of T_k. All subdivision sets are recorded as T_k^+. **Step 4: Lower bound** For each $M \subset T_k^+$, if *M* is infeasible, $\beta(M) = +\infty$. If *M* is feasible, $\beta(M) \leq \inf f(M \cap D)$. If *M* is uncertain, $\beta(M) \leq \inf f(M)$. Here, $\forall M' \in \mathsf{M}_{k-1}$, when $M' \supset M \in T_k^+$, $\beta(M) \geq \beta(M')$. **Step 5: Upper bound** Define $\mathsf{M}_k^+ = \{M \in T_k^+ \mid \beta(M) < \alpha_{k-1}\}$. For any $M \in \mathsf{M}_k^+$, confirm feasible point set $S(M) \subseteq M \cap D$. Here, $\forall M' \in \mathsf{M}_{k-1}$, when $M' \supset M \in \mathsf{M}_k^+$, $S(M) \supseteq M \cap S(M')$. $\alpha(M) = \min\{f(S(M)), +\infty\}$. **Step 6: Confirm unknown subdivision sets** Define $\mathsf{M}_k = (Q_k \setminus T_k) \cup \mathsf{M}_k^+$ and calculate $\alpha_k = \min\{\alpha(M) \mid M \in \mathsf{M}_k\}$, $\beta_k = \min\{\beta(M) \mid M \in \mathsf{M}_k\}$. If $\alpha_k < +\infty$, choose $\boldsymbol{x}^k \in D$ fitted with $f(\boldsymbol{x}^k) = \alpha_k$, $k+1 \to k$. Turn to step 2. End.

Combination algorithm

For the convenience of description, record the integer non-linear programming problem (5) and (6) as A, and the corresponding relaxation non-linear programming problem as B.

Step 1: Use the sequence quadratic programming method to solve B. If there is no feasible solution for B, stop calculating. If there is an optimal solution for B that conforms to the integer condition of problem A, stop calculating. If there is an optimal solution for B that doesn't conform to the integer condition of problem A, record its objective function value as $\underline{f}(\boldsymbol{x})$. **Step 2:** Find out an integer feasible solution of A. $x_j = 1,\ j = 1,2,...,n$. Record the objective function value as $\overline{f}(\boldsymbol{x})$. Define $f^*(\boldsymbol{x})$ as the optimal value, then $\underline{f}(\boldsymbol{x}) \leq f^*(\boldsymbol{x}) \leq \overline{f}(\boldsymbol{x})$. **Step 3:** Choose a random variable x_j from the optimal solution of B, which doesn't conform to the integer condition. $x_j = a_j$. Construct two constraints, $x_j \leq [a_j]$ and $x_j \geq [a_j]+1$. Then two subsequent planning problem B_1 and B_2 can be got. Regardless of the integer constraints, solve the B_1 and B_2 by SQP method respectively. **Step 4:** Consider each subsequent subproblem as a branch and indicate the solving results. Comparing with other solution, define the minimum optimal objective function value as a new lower bound $\underline{f}(\boldsymbol{x})$. Find out the minimum objective function value from each branch which is fitted with the integer condition, and define it as a new upper bound $\overline{f}(\boldsymbol{x})$. If the solution doesn't meet the requirements for the integer, keep the original upper bound values. **Step 5:** If there is a branch greater than $\overline{f}(\boldsymbol{x})$ in all branches optimal objective functions, cut off the branch without considering any longer. If there is a branch less than $\overline{f}(\boldsymbol{x})$ and not fitted with the integer condition, repeat the step 3, until $f^*(\boldsymbol{x}) = \overline{f}(\boldsymbol{x})$. End.

4. Numerical example

The logistics service supply chains are composed of functional logistics service vendors, logistics services integrator and logistics services user. Here functional logistics service vendors include TSV, SSV, DSV and OSV. All members provide independent logistics services.

The reliability and cost indicators of same type functional logistics service vendors are shown in table 1. For each type, the reliability and cost indicators of different vendors are same.

The minimum reliability of logistics service supply chain is 0.85. In order to minimize the total cost of logistics service supply chain, it needs to determine the number of all kinds of logistics service vendors.

Using Matlab7.0 programming, initial feasible solution is given. Here, $x_1 = 1$, $x_2 = 1$, $x_3 = 1$, $x_4 = 1$. By SQP algorithm, we get the optimal solution without considering the integer constraints. Then, $x_1 = 1.5815$, $x_2 = 1.7721$, $x_3 = 1.7154$, $x_4 = 1.4610$. The iterations are 18. The function counts are 122. Then branch, bound and prune. Add constraint conditions after each branch and calculate by SQP algorithm. Repeat the process until it gets the integer optimal solution. $x_1 = 2$, $x_2 = 2$, $x_3 = 3$, $x_4 = 1$. $f(x) = 6620$. The reliability of the logistics service supply chain is 0.809. The cycles are 164.

Tab.1 The related indicators of functional logistics service vendors

Types	Reliability	Cost
TSV	0.845	850
SSV	0.783	900
DSV	0.827	750
OSV	0.874	870

Tab.2 The optimal and near optimal solution

x_1	x_2	x_3	x_4	f	R	note
2	2	3	1	6620	0.809	optimal
3	2	2	1	6720	0.805	near optimal
2	2	2	2	6740	0.888	near optimal
2	3	2	1	6770	0.819	near optimal

To maintain the reliability, the logistics services integrator will choose 2 transportation service vendors, 2 storage service vendors, 3 distribution service vendors and 1 other service vendor. It is the optimal solution. Table 2 gives part of other near optimal solutions.

CONCLUSION

This study used an integer non-linear programming model in solving functional logistics service vendors selection problem. The paper built a combination algorithm based on improved sequential quadratic programming and branch and bound method. By numerical simulation, the paper got the following results: the optimal number of transportation service vendors was 2, the optimal number of storage service vendors is 2, the optimal number of distribution service vendors is 3, and the optimal number of other service vendors is 1. The minimum cost of logistics service supply chain was 6620 and the reliability was 0.809.The further study of the current work can be focus on increasing the objectives and constraints to make the model more robust. For example, detail the cost of vendors, especially the coordination cost. With the complication of the problem, it will produce more branches.

Acknowledgements

This work was supported by China Postdoctoral Science Foundation (No. 2014M551526), the Social Science Foundation of Jiangsu Province, China (No. 11SHD011), the Postdoctoral Research Foundation of Jiangsu Province, China (No. 1302097B) and the Senior Talents Foundation of Jiangsu University, China (No. 11JDG119). We thank the experts and anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] Tam MCY, Tummala VMR. An application of the AHP in vendor selection of a telecommunications system. Omega **2001**;29:171-182.

[2] Handfield R, Walton SV, Sroufe R, Melnyk SA. *European Journal of Operational Research* **2002**;141:70-87.

[3] Peric T, Babic Z, Veza I. *Int J Comp Integ M*, **2013**; 26:816-829.
[4] Hsu P, Lan K, Tsai C. *Int J Enter* S, **2013**;9:62-75.
[5] Degraeve Z, Labro E, Roodhooft F. *European Journal of Operational Research* **2000**;125:34-58.
[6] Zhang JL, Chen J. *Computers and Operations Research* **2013**; 40:2703-2710.
[7] Ghodsypour SH, O Brien C. *Int J Prod Econ*, **2001**;73:15-27.
[8] Hsu CH, Wang FK, Tzeng GH. *Resources Conservation and Recycling* **2012**; 66:95-111.
[9] Yu YG, Hong ZF, Zhang L, Chu CB. *European Journal of Operational Research* **2013**;225:273-284.
[10] Weber CA, Current JR, Desai A. *European Journal of Operational Research* **1998**;108:208-223.
[11] Wang G, Huang SH, Dismukes JP. *International Journal of Production Economics* **2004**;91:1-15.
[12] Qian SD. Operational research (revised edition). Beijing: the Tsinghua university press. **2001**.
[13] Ighravwe DE, Oke SA. *International Journal of Production Economics.* **2014**;150:204-214.
[14] Emam OE. *Applied Mathematics and Computation* **2006**; 172:62-71.
[15] Murty K, Kabadi S. *Mathematical Programming* **1987**;39:117-129.
[16] Huang HX, Han JY. Mathematical programming. Beijing: the Tsinghua university press. **2006**.