Multi-criteria logistics distribution network design using SAS/OR

William Ho *, Ali Emrouznejad

Operations and Information Management Group, Aston Business School, Aston University, Birmingham B4 7ET, United Kingdom

Abstract

This paper explores the use of the optimization procedures in SAS/OR software with application to the contemporary logistics distribution network design using an integrated multiple criteria decision making approach. Unlike the traditional optimization techniques, the proposed approach, combining analytic hierarchy process (AHP) and goal programming (GP), considers both quantitative and qualitative factors. In the integrated approach, AHP is used to determine the relative importance weightings or priorities of alternative warehouses with respect to both deliverer oriented and customer oriented criteria. Then, a GP model incorporating the constraints of system, resource, and AHP priority is formulated to select the best set of warehouses without exceeding the limited available resources. To facilitate the use of integrated multiple criteria decision making approach by SAS users, an ORMCDM code was implemented in the SAS programming language. The SAS macro developed in this paper selects the chosen variables from a SAS data file and constructs sets of linear programming models based on the selected GP model. An example is given to illustrate how one could use the code to design the logistics distribution network.

Keywords:
SAS
SAS/OR
Multiple criteria decision making
Analytic hierarchy process
Goal programming
Logistics distribution network

1. Introduction

The logistics distribution problem is to allocate a number of points of consumption to a number of points of supply, including suppliers, manufacturers, warehouses, distribution centers, and customers. The connection of these various logistics stakeholders by a mean of transportation facilities is regarded as the logistics distribution network. Logistics distribution network design is one of the major decision problems arising in contemporary supply chain management. There are mainly two inadequacies in the traditional approaches for the problem. First, a single criterion was focused only. The objective was either to minimize the total logistics cost (Hwang, 2005; Su, 1998; Wasner & Zäpfel, 2004) or total delivery time (Su, 1999). Second, only quantifiable data were considered in the optimization techniques. Some qualitative factors, which are mainly customer oriented, were not considered.

Multiple criteria decision making (MCDM) techniques have been used in recent years. One of the most prevalent techniques is analytic hierarchy process (AHP) (Ho, 2008). Some researchers (Korpela & Lehmsvaaara, 1999; Korpela, Lehmsvaaara, & Tuominen, 2001a, Korpela, Kyläheiko, Lehmsvaaara, & Tuominen, 2001b, 2002) applied the combined AHP-mixed integer linear programming (MILP) model approach for the network design, whereas another group of researchers (Chan & Chung, 2004a, 2004b, 2005; Chan, Chung, & Wadhwa, 2004, 2005, 2006) applied the combined AHP-genetic algorithm (GA) approach to solve the problem. For the combined AHP-MILP approach, the selection of distribution network was simply based on the customer satisfaction priorities instead of minimizing the total logistics cost or maximizing the total profit. Therefore, it is believed that the selected distribution network may not be cost effective. For the combined AHP-GA approach, the evaluation criteria used in AHP are all quantitative such as total cost, total delivery day, effectiveness of capacity utilization for warehouses, and so on. Some qualitative factors such as flexibility of capacity and value-added services were neglected. These factors are crucial in the integrated logistics system because they affect the customer satisfaction directly.

To overcome the drawbacks, this paper develops a systematic and prominent MCDM technique, combining AHP and goal programming (GP), to design an optimal logistics distribution network. The combined AHP-GP approach considers both quantitative and qualitative factors and also aims at maximizing the benefits of deliverer and customers.

SAS is recognized as one of the lead packages for statistical analysis and as a powerful tool for data base systems in many organizations, both in public and private sectors. SAS users come from every major industry (banking to pharmaceuticals, manufacturing to telecommunications, and so on). All with the same basic needs to make better strategic decisions and to gain a competitive edge (Emrouznejad, 2005).

There are many applications in SAS that the users recognized as powerful tools in organizational management. For example, the SAS/OR system has numerous optimization procedures which handle the standard problems such as linear programming (LP) and nonlinear programming (NLP) with all types of constraints. In
addition to the standard procedures available in SAS system applications such as neural network, simulation, and control project management are introduced.

This paper aims to introduce a new application in SAS system for yielding an optimal solution to a multi-criteria logistics distribution problem by solving the combined AHP-GP model.

2. MCDM techniques

MCDM techniques are generally divided into two categories: multiple attribute decision making (MADM) and multiple objective decision making (MODM). MADM techniques aim at selecting the best solution from a population of feasible alternatives which characterized by multiple attributes. One of the commonly used MADM techniques is AHP (Saaty, 1980). MODM techniques are a special extension of linear programming. A model is defined as a linear programming when the single objective function and the constraints involve linear expressions, and the decision variables are continuous. But, in MODM techniques, multiple objective functions are incorporated into the model simultaneously. GP (Charnes & Cooper, 1961) is an example of the MODM techniques.

In this paper, we apply AHP to evaluate the relative importance weightings of alternative warehouses with respect to some qualitative-based evaluating criteria. Then, a GP model is formulated to select the optimal set of warehouses while considering the AHP priorities of warehouses and the quantitative-based limitations of resources.

2.1. Prioritization of warehouses using AHP

In this paper, five criteria are proposed to evaluate the performance of warehouses. They include total lead time, reliability of order fulfillment, quality, flexibility of capacity, and value-added services. Total lead time comprises the time of handling inventory in warehouses, the time of storing/loading inventory in warehouses, and the time of delivering products from warehouses to customers. Reliability of order fulfillment consists of the accuracy of quantity fulfillment, the accuracy of due date fulfillment, and reliability of delivery time. Quality involves the commitment of deliverer to provide high-quality products and the condition of products received by customers. Flexibility of capacity refers to the ability of warehouses to respond to fluctuation in volume of customer orders. Value-added services refer to any activities that facilitate customers (e.g. track-and-trace and 24-hour customer hotline) and the responsiveness of warehouses to customer special requests (e.g. secure packaging and urgent delivery).

The first step of AHP for evaluating the performance of warehouses is to develop a hierarchy of the decision problem as shown in Fig. 1. After constructing the hierarchy, two criteria are compared at a time with respect to the goal. Once the pairwise comparisons have been made for the five criteria, each alternative warehouse is compared against each other alternative with respect to the corresponding criterion at a time. After completion of all pairwise comparisons, the relative priority of each criterion (from Table 1) and each alternative (from Table 2) are synthesized. The judgments are acceptable because the consistency ratios are all below the maximum 0.10 level. The overall priority ranking of warehouses is: wp1 = 0.446, wp2 = 0.180, wp3 = 0.239, and wp4 = 0.135. The AHP priorities are used to determine the priority level of the AHP priority constraints in the GP model.

2.2. Logistics distribution network design in GP

Consider a typical logistics distribution network which consists of m warehouses denoted as i = {1, 2, . . ., m} and n customers denoted as j = {1, 2, . . ., n}. Each warehouse has a maximum throughput, Qi, a minimum throughput, qi, a fixed cost, fc i, and a unit inventory holding cost, hic i. Each customer has a unique order volume, Dj. When warehouse i is assigned to serve customer j, it costs dcij dollars per unit for delivery. If the total amount of products assigned to warehouse i (i.e. \( \sum x_{ij} \)) is less than qi, this is regarded as impractical allocation because it is not cost-effective to set up a warehouse for processing only a few orders. To avoid low effectiveness of warehouse utilization, penalty cost, pci, is considered in the cost function.
model, which is incurred if 0 < \sum x_i < q_i. The problem here is to determine an optimal distribution network, which refers to the allocation of orders to the best warehouses. In the model, there are four types of decision variables:

\( x_i \) = amount of products delivered from warehouse \( i \) to customer \( j \)

\( u_i = \begin{cases} 
1 & \text{if total allocation of products to warehouse } i \text{ is less than } q_i \\
0 & \text{otherwise}
\end{cases} 
\)

\( v_i = \begin{cases} 
1 & \text{if warehouse } i \text{ is selected} \\
0 & \text{otherwise}
\end{cases} 
\)

\( w_i = \begin{cases} 
1 & \text{if both } u_i \text{ and } v_i \text{ equal to one} \\
0 & \text{otherwise}
\end{cases} 
\)

In the model, there are three types of constraints: system constraints, resource constraints, and AHP priority constraints. System constraints are ordinary linear programming constraints, in which there is no deviation variable. This type of constraints cannot be violated, and thus they are called hard constraints. Resource constraints are goal equations or soft constraints, in which there are deviation variables. AHP priority constraints are akin to resource constraints. In this type of constraints, there are deviation variables of which the priority levels are dependent on the overall AHP priority ranking.

**Standard multi-criteria logistics distribution model**

**System constraints**

\[ \sum v_i = m \]  

\[ \sum x_i + Mu_i \geq q_i \quad \forall i \]  

\[ \sum x_i - Mu_i \leq 0 \quad \forall i \]  

\[ w_i - u_i - v_i = -1 \quad \forall i \]  

**Resource constraints**

Priority 1 (\( P_1 \)):

\[ \sum x_i - d_i^+ + d_i^- = Q_i \quad \forall i \]  

Priority 2 (\( P_2 \)):

\[ \sum_i (xc_i + dc_i)x_i + \sum_i fc_i v_i - d_i^+ + d_i^- = TC \]  

Priority 3 (\( P_3 \)):

\[ \sum_i p_i c_i w_i - d_i^+ + d_i^- = 0 \]  

AHP priority constraints

Priority 4 to 4 + \( m - 1 \) (\( P_4 \) to \( P_{4+m-1} \)):

\[ v_i - d_i^+ + d_i^- = 1 \quad \forall i \]  

**Objective function**

Minimize \( z = \sum_k p_k (d_i^+ + d_i^-) + \sum_k p_k (d_i^+ + d_i^-) \) \( \forall r, s \)  

Constraint set (1) ensures that the number of warehouses selected must be equal to or less than the number of warehouses available. Constraint set (2) determines which warehouse(s) has/have allocation of products that less than minimum warehouse throughput. Constraint set (3) determines which warehouse(s) is/are selected. Constraint set (4) determines which warehouse(s) incur(s) penalty cost. Constraint set (5) allocates products to warehouses while the amount must not exceed the maximum warehouse throughput. Constraint set (6) allocates products to warehouses while the amount must equal to that demanded by the customers. Constraint set (7) ensures that the total cost, including the inventory holding cost, delivery cost, and fixed cost associated with warehouse selection, must not exceed the targeted amount. Constraint set (8) ensures that allocation of products to warehouses incurring penalty cost is not allowed. Constraint set (9) is to select warehouse \( i \). Objective function (10) is to minimize the total deviations from the goals.

3. **ORMCDM: SAS code for multi-criteria logistics distribution network design**

The ORMCDM introduced in this paper provides a powerful decision making tool for yielding an optimal logistics distribution network design. To enhance the model there are several parameters. The user can select the desired parameters according to the particular model that is required. Users familiar with SAS can add their own features. Users not familiar with SAS need only to run the program with their model specification prior to running the system.

The ORMCDM requires an initial data set that contains the name of variable and data file for warehouses and customers. The data is saved as SAS data set. SAS has the ability also to read from a text (tab delimited) or excel format. The program has the
ability to accommodate unlimited number of warehouses and customers. The only limitation is the memory of computer used to run the ORMCDM.

The ORMCDM software then converts data set to a GP model. Based on the data and parameters specified in the ORMCDM, the code first creates the usual linear program, then use "PROC LP" to solve the model. The results will then be transferred to report files. The ORMCDM produces a table listing values of decision variables and deviation variables. It also supplies a report summarizing the status of goal achievement at each stage (or priority level).

In the rest of this paper the procedure of implementation of ORMCDM together with an example are explained. Fig. 2 illustrates the data flow in the ORMCDM.

4. Definition of terms and typographical conventions

In the rest of this paper and particularly in the ORMCDM code we will see several types of styles used. Style conventions are summarized below:

- Courier font: is used to show example of SAS statements. In most cases, this paper uses lowercase type for SAS code. SAS users can enter their own SAS code in lowercase, uppercase or a mixture of the two. Enter any titles and footnotes exactly as you want them to appear on the printout.
- _Underscore_: variable name that are surround by "_" are specifically used as parameters to the SAS. In all case these variables must be used without any change.
- _Underscore: variable name that are started with "_" are specifically used as parameters to the ORMCDM.

The ORMCDM runs four macros for data handling (%data), model building (%model1 and %model2), and report writing (%report).

5. Illustration of ORMCDM

This section presents a simple example of four warehouses and seven customers for illustration of ORMCDM. The resource data includes:

- QQi: maximum throughput of warehouse i.
- qi: minimum throughput of warehouse i.
- hci: unit inventory holding cost associated with warehouse i.
- fci: fixed cost associated with warehouse i.
- pci: penalty cost associated with warehouse i.
- cost: unit delivery cost from warehouse i to customer j.
- demand: unique order volume requested by the customers.

6. Data handling (%data)

This part of ORMCDM prepares the data to a suitable format that can be used in PROC LP. The ORMCDM requires one data set containing name of “warehouses”, name of “customers”, “cost matrix”, and “demand”. The data set should be a “.txt” file, which is saved as “text tab delimited”. The warehouses’ and customers’ names must start with a letter and may contain up to 50 characters. The variable names must be listed in the first row of the data file. The warehouses’ names should be listed in the first column. Note that the demand should be given in the last row of the data set. Other information, including QQi, qi, hci, fci, and pci should also be given in the data set, but the order of the columns is not important. An example of data file is:

File: data.txt

There is only one parameter prior to calling the data macro:

- _data: indicates the name and location of the data file.

SAS procedure for data handling is presented in Appendix A.1.

7. Model building (%model)

This part of ORMCDM calls PROC LP to solve the model. There are two SAS macros for model building. Macro %model1 creates a suitable data set that can be used for PROC LP. There are four parameters prior to calling the model macro:

- _nWareh: defines the number of warehouses.
- _nCustomer: defines the number of customers.
• _TotalCost: defines the total cost.
• _myupperbnd: defines the upper bound value for integer variables.

Macro %model2 defines seven data sets that could be used to solve the multi-criteria logistics distribution problem in seven stages. At each stage, a data set is created, then PROC LP solves the model and results are saved in three data sets for future analysis. These three output data sets are for primal solution, dual solution, and tabulated solution of each stage. There are two parameters before calling this macro.

• _myLarge: defines an optional large number.
• _nIterations: defines an optional value for number of integer iterations.

The parameter _myLarge is an optional parameter that is needed for LP, and in this case, we set it to 100000.

SAS procedure for models %model1 and %model2 building are presented in Appendix A.2, A.3, respectively.

8. Report writing (%report)

The outputs from ORMCDM include one report for each stage. This report contains all information regarding the summary and solution of each stage of the problem, in this case, seven stages. All these information are also saved in SAS data sets. The users can define appropriate names for each of these data sets before calling %ormcdm macro as follows.

• _outprimal: identifies the name of the SAS output file for primal solution.
• _outdual: identifies the name of the SAS output file for dual solution.
• _outtable: identifies the name of the SAS output file for tabulated solution.

![Fig. 3. Log for %ORMCDM.](image-url)
Another parameter needs to be set before calling this macro is the \_title as follows:

- \_title: gives a title in the output of the SAS.

SAS procedure for report writing is presented in Appendix A.4.

9. ORMCDM macro (%ormcdm)

To make the system as ease as possible, the \"%ormcdm macro\" put all the above code together.

\begin{verbatim}
\%macro ormcdm;
\%data;
\%model1;
\%do i=1 %to &nWareh+3;
* Number of stages in GP model;
\%model2(&i);
\%report(&i);
\%end;
\%mend ormcdm;
\end{verbatim}

\end{verbatim}

\begin{verbatim}
\begin{tabular}{|c|c|c|c|c|c|}
\hline
VAR & TYPE & STATUS & LB & VALUE & UB & R Cost \\
\hline
x1 & INTEGER & BASIC & 944 & 0 & 0 & 0 \\
\hline
x12 & INTEGER & BASIC & 900 & 0 & 0 & 0 \\
\hline
x13 & INTEGER & BASIC & 1000 & 0 & 0 & 0 \\
\hline
x14 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x15 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x16 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x17 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x18 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x19 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x20 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x21 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x22 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x23 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x24 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x25 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x26 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x27 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x28 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x29 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x30 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x31 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x32 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x33 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x34 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x35 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x36 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x37 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x38 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x39 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x40 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x41 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x42 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x43 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x44 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x45 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x46 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
x47 & INTEGER & BASIC & 0 & 0 & 0 & 0 \\
\hline
1 & BINARY & BASIC & 1 & 0 & 0 \\
\hline
u & BINARY & BASIC & 0 & 0 & 0 \\
\hline
1 & BINARY & BASIC & 0 & 0 & 1 \\
\hline
2 & BINARY & BASIC & 0 & 0 & 1 \\
\hline
3 & BINARY & BASIC & 0 & 0 & 1 \\
\hline
4 & BINARY & BASIC & 0 & 0 & 1 \\
\hline
5 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
6 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
7 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
8 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
9 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
10 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
11 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
12 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
13 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
14 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
15 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
16 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
17 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
18 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
19 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
20 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
21 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
22 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
23 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
24 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
25 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
26 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
27 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
28 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
29 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
30 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
31 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
32 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
33 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
34 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
35 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
36 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
37 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
38 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
39 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
40 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
41 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
42 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
43 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
44 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
45 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
46 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
47 & BINARY & BASIC & 0 & 0 & 0 \\
\hline
\hline
\end{tabular}
\end{verbatim}

Fig. 4. Result of ORMCDM, primal output of stage 5.

Fig. 5. Result of ORMCDM, primal output of stage 6.
In the above code, the “%ORMCDM macro” is used to manage all previously explained codes, including data handling, model building, and report writing. To get the result, user needs to set up the parameters and run the following code:

```
%ORMCDM;
```

### 10. Instruction of how to use ORMCDM macro

This section presents SAS code for the earlier example of the multi-criteria logistics distribution problem with four warehouses and seven customers. The data is saved in file “data.txt”.

A user needs to set the parameters as required and run the following code.

```
options nodate;
%let _title='An example of MCDM';
%let _dataMCDM='c:/sasor/data.txt';
%let _data='c:\sasor\MCDM.txt';
%let nWareh=4;
%let nCustomer=7;
%let _TotalCost=425000;
%let _outprimal=outprimal;
%let _outdual=outdual;
%let _outtable=outtable;
%let _myLarge=100000;
%let _niterations=100000;
%let _myupperbnd=12000;
%ormcdm;
```

The above code manages to get the result based on the specified parameters and the cost matrix saved in the text file. The above code also produces a macro variable (._ORLP_) at termination. Users can examine the result of this macro variable, examine whether PROC LP ran correctly, and examine what error or difficulty it encountered. Summary of information, including the objective value at optimum level and status of _ORLP_, can be seen in the log file as shown in Fig. 3.

### 11. Sample results from ORMCDM macro: output from SAS

In the GP model, there are 40 integral decision variables, 34 deviation variables, 30 constraints, and seven goals (or stages). Some of the results of running the above code are presented in Figs. 4 and 5, which summarize the solutions in stage 5 and stage 6, respectively. The solution, shown in Fig. 4, is feasible because the allocation does not exceed the maximum throughput of warehouses, does satisfy the volume requirement of customers, does not exceed the total cost budget, and does not incur any penalty cost. When stage 6 was found to be unachievable (i.e. $d_{N16} = 1$), shown in Fig. 5, the optimization process was terminated. So, the solution, satisfying the first five priority levels (i.e. $P_1$ to $P_5$), is an optimal solution of the problem, shown in Fig. 4. The values of decision variables $\theta_i$ show that three warehouses were selected, including warehouse 1 ($\theta_1 = 1$), warehouse 3 ($\theta_3 = 1$), and warehouse 4 ($\theta_4 = 1$). The total cost spent in setting up these three warehouses, holding inventory in the warehouses, and delivering products from the warehouses to their assigned customers is $425000 with no slack. Besides, the total penalty cost incurred is zero.

### 12. Conclusions

Nowadays, there is a plenty of software that can be used for operations research. The major reason why we selected SAS is because it has various optimization tools that can be used in a wide range of problems in operations research. In particular, optimization procedures in SAS/OR are exposed to the user in a variety of places such as “PROC LP” and “PROC NLP”. ORMCDM as introduced in this paper is a new application in SAS/OR that is a powerful decision making tool for designing an optimal logistics distribution network. The ORMCDM application implemented in this paper has no limitation on the number of nodes (warehouses and customers). The only limitation is the memory and disk space of the computer used. Besides, SAS has strong data management capabilities that can handle very large datasets efficiently, and it can work with multiple datasets simultaneously. In cases where SAS users want to solve another problem or change the data, they can simply revise a single data set file and run a single statement, %ormcdm. The ORMCDM report’s files can also directly feed to other SAS routines for further analysis.

### Appendix A

#### A.1. SAS procedure for data handling

```
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
* The data handling macro;
%macro model1;
%macro data;
%macro model1;
* The model building macro;
end;
myst = '_row_';
put myst @@;
myst0 = 'X';
do i = 1 to &_nWareh;
do j = 1 to &_nCustomer;
ds = myst0||put(i,1.)||put(j,1.);
put ds @@;
end;
end;
do i = 1 to &_nWareh;
ds = 'w'||put(i,1.); put ds @@;
ds = 'u'||put(i,1.); put ds @@;
ds = 'v'||put(i,1.); put ds @@;
end;
do i = 1 to 2*&_nWareh + &_nCustomer + 2;
if i < &_nWareh + &_nCustomer then mycoefP = 1;
else mycoefP = 0;
if &_nWareh < i <= &_nWareh + &_nCustomer then myco-
efN = 1; else mycoefN = 0;
put '. ' @@;
put mycoef @@;
end;

myst = 'binary';
put myst @@;
myInt = 0;
do i = 1 to &_nWareh;
do j = 1 to &_nCustomer;
put '. ' @@;
end;
do i = 1 to &_nWareh;
myInt = myInt + 1;
put myInt @@;
myInt = myInt + 1;
put myInt @@;
myInt = myInt + 1;
put myInt @@;
end;
do i = 1 to 2*&_nWareh + &_nCustomer + 2;
if i <= &_nWareh + &_nCustomer then mycoefP = 1;
else mycoefP = 0;
if &_nWareh + 1 <= &_nWareh + &_nCustomer then myco-
efN = 1; else mycoefN = 0;
put '. ' @@;
put mycoefN @@;
end;

myst = 'integer .';
put myst @@;

myInt = 0;
do i = 1 to &_nWareh;
do j = 1 to &_nCustomer;
put '. ' @@;
end;
do i = 1 to &_nWareh;
myInt = myInt + 1;
put myInt @@;
myInt = myInt + 1;
put myInt @@;
myInt = myInt + 1;
put myInt @@;
end;
do i = 1 to 2*&_nWareh + &_nCustomer + 2;
if i <= &_nWareh + &_nCustomer then mycoefP = 1;
else mycoefP = 0;
if &_nWareh + 1 <= &_nWareh + &_nCustomer then myco-
efN = 1; else mycoefN = 0;
put '. ' @@;
put mycoefN @@;
end;

myst = 'upperbound';
put myst @@;

myInt = 0;
do i = 1 to &_nWareh;
do j = 1 to &_nCustomer;
myUpp = &_myupperbnd;
put myUpp @@;
end;
do i = 1 to 2*&_nWareh + &_nCustomer + 2;
if i <= &_nWareh + &_nCustomer then mycoefP = 1;
else mycoefP = 0;
if &_nWareh + 1 <= &_nWareh + &_nCustomer then myco-
efN = 1; else mycoefN = 0;
put '. ' @@;
put mycoefN @@;
end;

myst = 'objP1';
put myst @@;
do i = 1 to &_nWareh;
do j = 1 to &_nCustomer;
put '0 ' @@;
end;
do i = 1 to &_nWareh;
put '0 ' @@;
put '0 ' @@;
put '0 ' @@;
end;
do i = 1 to 2*&_nWareh + &_nCustomer + 2;
if i <= &_nWareh + &_nCustomer then mycoefP = 1;
else mycoefP = 0;
if &_nWareh + 1 <= &_nWareh + &_nCustomer then myco-
efN = 1; else mycoefN = 0;
put '. ' @@;
put mycoefN @@;
end;

myst = 'objP2';
put myst @@;
do i = 1 to &_nWareh;
do j = 1 to &_nCustomer;
put '0 ' @@;
end;
do i = 1 to &_nWareh;
put '0 ' @@;
put '0 ' @@;
put '0 ' @@;
end;
do i = 1 to 2*&_nWareh + &_nCustomer + 2;
if i = &_nWareh + &_nCustomer + 1 then coef = 1;
else coef = 0;
put '0 ' @@;
put mycoef @@;
end;

myst = 'upperbd .';
put myst @@;

myst = 'objP3';
put myst @@;
do i = 1 to &nWareh;
do j = 1 to &nCustomer;
put '0' @@;
end;
end;
do i = 1 to &nWareh;
put '0' @@;
put '0' @@;
put '0' @@;
end;
do i = 1 to 2* &nWareh + &nCustomer + 2;
if k <= &nWareh + &nCustomer then myst1 = 'P1';
else if k <= &nWareh + &nCustomer + 1 then myst1 = 'P2';
else if k <= &nWareh + &nCustomer + 2 then myst1 = 'P3';
else myst1 = 'P'||put(k-(&nCustomer + 3),1.);
if k<10 then myst = 'dZrP'||myst1||put(k,1.);
else myst = 'dZrP'||myst1||put(k,2.);
put myst @@;
do i = 1 to &nWareh;
do j = 1 to &nCustomer;
push '0' @@;
end;
end;
do i = 1 to 2* &nWareh + &nCustomer + 2;
if i = &nWareh + &nCustomer + 2 + k then mycoef = 1;
else mycoef = 0;
put mycoef @@;
end;
end;
doi = 1 to 2* &nWareh + &nCustomer + 2;
if i = &nWareh + &nCustomer + 2 + k then mycoef = 1;
else mycoef = 0;
put mycoef @@;
end;
end;
doi = 1 to &nWareh;
put '0' @@;
end;
end;
doi = 1 to &nWareh;
put '0' @@;
end;
end;
edo i = 1 to &nWareh;
put '0' @@;
end;
end;
odk = 1 to &nWareh;
put myst @@;
doi = 1 to &nWareh;
do j = 1 to &nCustomer;
pusht '0' @@;
end;
end;
edo i = 1 to &nWareh;
put '0' @@;
end;
end;
edo i = 1 to 2* &nWareh + &nCustomer + 2;
if i = k then mycoef = 1; else mycoef = 0;
pusht '0' @@;
end;
edo i = 1 to 2* &nWareh + &nCustomer + 2;
if i = k then mycoef = 1; else mycoef = 0;
pusht '0' @@;
end;
edo k = 1 to 2* &nWareh + &nCustomer + 2;
if k < &nWareh + &nCustomer then myst1 = 'P1';
else if k < &nWareh + &nCustomer + 1 then myst1 = 'P2';
else if k < &nWareh + &nCustomer + 2 then myst1 = 'P3';
else myst1 = 'P'||put(k-(&nCustomer + 3),1.);
if k<10 then myst = 'dZrP'||myst1||put(k,1.);
else myst = 'dZrP'||myst1||put(k,2.);
put myst @@;
do i = 1 to &nWareh;
do j = 1 to &nCustomer;
pusht '0' @@;
end;
end;
edo i = 1 to &nWareh;
put '0' @@;
end;
end;
edo i = 1 to &nWareh;
put '0' @@;
end;
end;
edo i = 1 to &nWareh;
put '0' @@;
end;
end;
edo i = 1 to 2* &nWareh + &nCustomer + 2;
if i = k then mycoefP = 1; else mycoefP = 0;
mycoefN = -mycoefP;
pusht '0' @@;
end;
end;
edo i = 1 to 2* &nWareh + &nCustomer + 2;
pusht '0' @@;
end;
end;
put 'eq -1' ;
end;
do k = 1 to &nWareh;
myst = 'XU'||put(k,1.);
poly mystical @ &nWareh;
do i = 1 to &nWareh;
if i = k then mycoef = 1; else mycoef = 0;
do j = 1 to &nCustomer;
poly mycoef @@;
end;
end;
do i = 1 to &nWareh;
if i = k then mycoef = &myLarge; else mycoef = 0;
poly '0' @@;
poly mycoef @@;
poly '0' @@;
end;
do i = 1 to 2*&nWareh + &nCustomer + 2;
poly '0' @@;
poly '0' @@;
end;
put 'ge @@
put '0'
end;
do k = 1 to &nWareh;
myst = 'XV'||put(k,1.);
poly myst @@;
do i = 1 to &nWareh;
if i = k then mycoef = 1; else mycoef = 0;
do j = 1 to &nCustomer;
poly mycoef @@;
end;
end;
do i = 1 to &nWareh;
if i = k then mycoef = -&myLarge; else mycoef = 0;
poly '0' @@;
poly mycoef @@;
poly '0' @@;
end;
do i = 1 to 2*&nWareh + &nCustomer + 2;
poly '0' @@;
poly '0' @@;
end;
put 'le @@
put '0'
end;
do k = 1 to &nWareh;
myst = 'XP1d'||put(k,1.);
poly myst @@;
do i = 1 to &nWareh;
if i = k then mycoef = 1; else mycoef = 0;
do j = 1 to &nCustomer;
poly mycoef @@;
end;
end;
do i = 1 to &nWareh;
if i = k then mycoef = &myLarge; else mycoef = 0;
poly '0' @@;
poly mycoef @@;
poly '0' @@;
end;
do i = 1 to 2*&nWareh + &nCustomer + 2;
poly '0' @@;
poly '0' @@;
end;
do k = 1 to &nWareh + 1 to &nCustomer + &nWareh;
if k < 10 then myst = 'Pld'||put(k,1.); else myst = 'Pld'||put(k,2.);
poly myst @@
do i = 1 to &nWareh;
do j = 1 to &nCustomer;
if j + &nWareh = k then mycoef = 1; else mycoef = 0;
poly mycoef @@;
end;
end;
do i = 1 to &nWareh;
poly '0' @@;
poly '0' @@;
end;
do i = 1 to 2*&nWareh + &nCustomer + 2;
if i = k then mycoef = 1; else mycoef = 0;
mycoefN = -mycoef;
poly mycoef @@
poly mycoefN @@;
end;
do i = 1 to &nWareh;
poly '0' @@
poly '0' @@
end;
put 'eq @@
poly demand[&nWareh]
end;
poly 'P2'
poly myst @@
do i = 1 to &nWareh
ndo j = 1 to &nCustomer
mycoef = cost[i,j] + hc[i];
poly mycoef @@
end;
end;
do i = 1 to &nWareh
poly '0' @@
poly '0' @@
poly fc[i] @@
end;
do k = 1 to &nWareh + 1 to &nCustomer + 2;
if i = &nWareh + &nCustomer + 1 then mycoef = 1; else mycoef = 0;
mycoefN = -mycoef;
poly mycoef @@
poly mycoefN @@;
end;
do i = 1 to &nWareh;
poly '0' @@
poly '0' @@
end;
do i = 1 to 2*&nWareh + &nCustomer + 2;
if i = &nWareh + &nCustomer + 1 then mycoef = 1; else mycoef = 0;
mycoefN = -mycoef;
poly mycoef @@
poly mycoefN @@
end;
do 'eq @@
poly mycost = &TotalCost;
poly mycost
end;
do 'P3'
poly myst @@
do i = 1 to &nWareh
ndo j = 1 to &nCustomer
poly '0' @@
end;
do i = 1 to &nWareh
poly '0' @@
poly pc[1] @@
end;
do i = 1 to 2*&nWareh + &nCustomer + 2;
if i = &nWareh + &nCustomer + 2 then mycoef = 1; else mycoef = 0;
mycoefN = -mycoef;
A.3. SAS procedure for model building (%model2)

```sas
put mycoef @@; put mycoefN @@; end;
put 'eq 0';
do k = 1 to &nWareh;
myP = 'objP'|put(&p,1.); if _type_ = 'min' and _row_ne maryP then delete;
if substring(_row_,1,5) = 'dZrNP' and substring(_row_,1,6) > 'dZrNP'|put(&p,1.); then delete;
if substring(_row_,1,5) = 'dZrPP' and substring(_row_,1,6) > 'dZrPP'|put(&p,1.); then delete;
run;
proc lp
   data = dataMCDM & p
dualout = &outdual & p
primalout = &outprimal & p
tableauout = &outtable & p
imaxit = &nIterations;
run;
%put &orlp; %end model2;
```

A.4. SAS procedure for report writing

```sas
* The report writing macro;
%macro report(p);
title '(P = ' & p ')';
proc print
   data = &outprimal & p;
proc print
   data = &outdual & p;
run;
%end report;
```

References


