Journal of Korean
Data & Information Science Society
2007, Vol. 18, No. 2, pp. 561 ~ 572

A Computationally Efficient Optimal Allocation Algorithms for Large Data[1)]

Il Hyung Kwon[2)] · Ju Sung Kim[3)]

Abstract

In this paper, we describe various efficient optimization algorithms for obtaining an optimal customer allocation in the telephone call center. The main advantages of the proposed algorithms are simple, fast and very attractive for massive dataset. The proposed algorithms also provide comparable performance with the other more sophisticated linear programming methods. The proposed optimal allocation algorithms increase the customer contact, response rate and management product and optimize the performance of call centers. Simulation results are given to demonstrate the effectiveness of our algorithms.

Keywords : Data Mining, Distance Method, Optimization, SAS Enterprise, Telephone Call Center

1. Introduction

The optimization problem is the branch of computational science that seeks the optimum as a numerical value. Such a problem arises in all areas of business, physical engineering, economics and management science. Optimal allocation algorithms are becoming an increasingly important means in all situations in which resources have to be allocated to categories. Optimal allocation has been solved by a number of operation researchers in various settings; see, for example, Dai et al. (2003), Berman and Cutler (2004), Kaspi and Shabtay (2004) and Shabtay (2004).

In customer relationship management (CRM) research, the researcher develops a model to allocate a number of agents, called staffing scheduling; Tanir and Booth

1) This work was supported by the grant of the Chungbuk National Univ. in 2006.
2) Researcher, 8-10F, 889-11, Daechi B/D, SAS Korea, 135-839 E-mail : stocks@hanmail.net
3) Corresponding Author: Professor, Department of Information and Statistics, Chungbuk National Univ., Cheongju, 360-763 E-mail : kimjs@chungbuk..ac.kr

(1999), Gulati and Malcolm(2001) and Pichitlamken, et al. (2003). Recently, Lee (2006) applied data mining method for inbound call center optimization to obtain the possibility of efficiency for the call center with the maximum efficiency.

In statistics, relatively recent activities have led to a large body of literature on a statistical analysis of telephone call centers using Poisson processes and focused on the arrival times of calls requesting service. These efforts typically have been modelled by inhomogeneous Poisson processes. A few key references are Brown, et al.(2002, 2005), Koole and Mandelbaum (2002), Brown and Shen (2002). Gans, et al. (2002) reviewed the state of research on telephone call centers and provided a tutorial on how call centers function and proceed to survey academic research devoted to the management of their operations.

While there is an abundance of research on this optimization, there is little research related to real world massive dataset applications. Optimization algorithms with massive dataset are difficult to develop because computer resources are limited. They require direct or indirect methods for obtaining the optimal value as the data increases, while guaranteeing that the value of objective function will have maximum. Although linear programming optimization methods may give us a correct answer, it has exponential running time and becomes too messy for models with many parameters, thus, we need to develop a new algorithm for large dataset to obtain an optimal allocation.

There are three major benefits afforded by the proposed algorithms. First, it is able to obtain the optimal or near-optimal solution without much computational effort for large dataset or even billions of records; the other algorithms such as simplex method are highly criticized for dealing with this issue. Second, the proposed algorithms are simple and fast but achieve a comparable performance with other algorithms. Third, the algorithm always uses a feasible initial values by the finding initial value algorithm; simplex methods sometimes fail to find the initial value.

In this paper, we consider the case of the optimal customer allocation problem in the telephone call center. This kind of optimal customer allocation problem has received only minimal attention in the telephone call center. We propose a simple optimal allocation algorithms to obtain the maximum value of an objective function.

The paper is organized as follows: Section 2 presents various optimal allocation algorithms. In Section 3, we compare the performance of the proposed algorithms and their agreement for massive dataset with the simulation results. Conclusions in Section 4 summarize the results and future research directions.

2. Optimal Allocation Algorithms: Call Center Application

In this section, we develop an approach that comprises a series of mathematical optimization formulations that solve the optimal allocation problem for massive

dataset. The goal of the proposed method is to find the optimal allocation to maximize the objective function considering the estimation precision and running time for massive dataset. They are direct methods and have a role to play where the data are massive. Design of the proposed algorithms certainly bring a new technique in the field of optimization of customer allocation for telephone call center.

To describe the optimization allocation problems, we introduce some notations. Suppose there are a collection of the customers $i, i=1,2,\cdots,n$ and feasible clusters $j, j=1,2,\cdots,m$ (e.g, time periods) which are classified to indicate variables. The optimal set of allocation of n customers can be described as the solutions to the optimization problem. Score matrix P represents indices, which is the predicted probability of successful contact of eligible customer based on prior customer history by logistic regression model using SAS Enterprise Miner. Denote the score of the i-th customer and j-th cluster by p_{ij} and the maximum number of allocated customers for each cluster expressed as $M=[\max_1, \max_2, \cdots, \max_m]$. It is assumed that $p_{ij} \in D$ where D is a set of nonnegative values which are less than or equal to 1. Letting the decision solutions $K_i, i=1,2,\cdots,n,$ represent the indicate vector of i-th customer expressed as $K_i = [k_{i1}, k_{i2}, \cdots, k_{im}]$, where k_{ij} is an indicator variable defined as

$$k_{ij} = \begin{cases} 1, & i \in cluster\ j \\ 0, & otherwise \end{cases}$$

and $\sum_{j=1}^{m} k_{ij} = 0$ or 1, where k_{ij} is a binary scheduling variable determining whether customer i will be assigned to cluster j or not. Therefore, we compute the optimal allocations where $count_j$ denotes the optimum number of assigned customers in cluster j providing maximum value of the objective function.

Using above notations, the various optimal allocation methods are stated as follows. A mathematical optimization problem consists of an objective function and a set of constraints expressed in the form of a system of equation or inequalities. The general optimization problem will be written as

$$\max_{K_i \in \Omega}\ Q = \sum_{i=1}^{n} f(K_i'),$$

where the objective function $Q: R^n \rightarrow R$ and subject to a set of constraints, and Ω is a space of R^n describing the possible decision values. If $K_i \in \Omega$, we call K_i a feasible solution and if the maximum of Q over $K_i \in \Omega$ occurs at $K_i = K_i^*$, we call K_i^* an optimal solution and $Q(K_i^*)$is the optimal value of the problem.

To solve the general optimization problem, simplex method and grid search method in linear programming generally are used for finding optimal solutions. Disadvantages of the simplex method are that it takes an exponential running time and the immense amount of numerical processing, theoretically. For the higher dimensional problems, the grid search method can be very slow because it requires a large number of grid points per iteration. When the objective function is not concave, grid may miss global maximum. Solving these problem for many parameter in optimization turned out to be extremely difficult. Thus, although there is rapid increase in available computing resources, it is impossible to find optimal allocation on massive dataset.

Therefore, we suggest algorithms which provide an near optimal solution close to optimal values without intensive computations. The proposed method, called the distance method, is based on Euclidean distance among the predicted probabilities of successful contact of eligible customer based on the prior customers history. One can think of the problem of allocating a number of customers among different clusters such that the solutions are optimal (e. g. in terms of high response rate).

2.1 Algorithm 1: Maximum Score Optimization Algorithm

Suppose that there are n customers that are assigned by high predicted probabilities in logistic model. Let the score matrix P be given or fixed and there are no restrictions of the maximum number of allocated customers for each cluster. Algorithm 1 procedure maximizes the objective function required for the score matrix P and decision solution K_i. In this algorithm, we consider the problem of assigning customers into each cluster so as to maximize the response rate. Let the highest score of i-th customer among m clusters in score matrix be $p_{i,\max} = \max(p_{i1}, p_{i2}, \cdots, p_{im})$, where $i = 1, 2, \cdots, n$. Suppose we need to find the optimal decision vector K_i^* out of a binary set $\Omega := \{0, 1\}$. A clever way to choose the optimal decision vector K_i as to maximize the objective function can be written as

$$\max_{K_i \in \Omega} Q = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{i,\max} \cdot k_{ij} \tag{1}$$

subject to $count_j \leq \max_j$ for $\forall j$ and $k_{ij} = \{0, 1\}$ for $\forall i, j$, where i is the calling the customer, j is the cluster, $p_{i,\max} \in R^n$ is a given score vector and k_{ij} is a binary scheduling variable. Let S_i be a selected cluster for i-th customer, then we assign the optimal allocation for m clusters as follows.

$$C = \left[\sum_{i=1}^{n} I(S_i = 1), \sum_{i=1}^{n} I(S_i = 2), \cdots, \sum_{i=1}^{n} I(S_i = m)\right]$$
$$= \left[\sum_{i=1}^{n} k_{i1}, \sum_{i=1}^{n} k_{i2}, \cdots, \sum_{i=1}^{n} k_{im}\right]$$
$$= [count_1, count_2, \cdots, count_m]$$

where $I(\bullet)$ is the indicate function. If there is no restriction of the maximum number of allocated customer, or the condition $count_j \leq \max_j$ for $\forall j$ is satisfied, we obtain the optimal value of the objective function (Q) same as using the other method in linear programming. The advantage of Algorithm 1 is that it can tremendously reduce the computational time to obtain the optimal value.

The algorithm 1 is useful only if there is no restriction of the maximum number of assigned customers or if the condition $count_j \leq \max_j$ is satisfied (i.e., the number of assigned customer is less than or equal to the maximum resources of a call center). Thus, it is not useful for the real-world problem because of this strict restrictions. However, this algorithm is easy to implement and reduces the computation time remarkably. The example above, although overly simple, indicates that the algorithm provide exact optimal solution. The optimization algorithm is formally stated as follows:

Algorithm 1 Maximum Score Optimization Algorithm Prototype

Step 1: Generate initial restrictions on $M = (\max_1, \max_2, \cdots, \max_m)$
Step 2: for $j = 1$ to n do
Step 3: Compute $p_{i,\max} = \max(p_{i1}, p_{i2}, \cdots, p_{im})$
Step 4: Compute K_i
Step 5: Compute S_i
Step 6: end for

2.2 Algorithm 2: Score Distance Optimization Algorithm for Only Two Clusters

More general problems are still opened. Although algorithm 1 provides the optimal solution, it has some limitations. Suppose available resources are sufficient for each cluster and greater than the number of customers. We then need to assign each customer into the most appropriate cluster to maximize the response rate. For simplicity, let us consider that we have only two clusters. In order to maximize the object function, we use the score distance between two clusters.

Algorithm 2, called the score distance optimization algorithm, provides exactly

same objective value Q as the true optimal value. Let the score matrix and score distance of two clusters be $P_{n\times 2}=\left[\hat{p_{i1}}\ \hat{p_{i2}}\right]$, $\hat{d_i}=\hat{p_{i1}}-\hat{p_{i2}}$, respectively. Let us define rank vector $\hat{r_i}=Rank(\hat{p_{i1}}-\hat{p_{i2}})$, $i=1,2,\cdots,n$.

Calls are then made in rank order throughout the customers. Ranking rules are based on the score from logistic regression modeling that based on a complex functions of prior customer attributes.

Using this algorithm, we assign the optimal allocation to two clusters

$$C=\left[\sum_{i=1}^{n} I(S_i=1)\quad \sum_{i=1}^{n} I(S_i=2)\right]=\left[count_1,\ count_2\right].$$

For fixed $m=2$, the algorithm presented below determines the optimal solution C provided by the maximum value of objective function. Algorithm 2 can be described formally as follows:

Algorithm 2 Scoring Distance Optimization Algorithm Prototype

Step 1: Generate initial restriction of $M=(\max_1, \max_2)$
Step 2: Compute the distance vector, D
Step 3: Compute the rank vector, R of distance vector
Step 4: for $i=1$ to n do
Step 5: if $r_i=\max_1$ and $d_i\geq 0$, then $S_i=1$
Step 6: else if $n-r_i+1\leq \max_2$ then $S_i=2$
Step 7: else if $n-r_i+1>\max_2$ then $S_i=1$
Step 8: end for

Algorithm 2 is simple but much faster than the other more complicated algorithms. The algorithm provides the optimal value of objective function same as the other method that has limitations such as running time and messy computation for massive dataset.

2.3 Algorithm 3: Iterated Score Distance Optimization Algorithm for More Than Two Clusters

The algorithm presented in the preceding section assumes that the number of cluster is at most two clusters. We propose a more complicated optimization algorithm with multiple clusters, say, $m>2$. A natural extension of the score

distance optimization algorithm is to solve the following optimization problem with multiple clusters which analogous to the preceding section. This algorithm uses the distance of scores for all possible combinations of two clusters to achieve the maximum value of objective function by iteration.

Suppose that the score matrix P is given and the number of cluster is greater than 2, $m > 2$. Algorithm 3, called the iterated score distance optimization algorithm, is the procedure that optimizes the objective function using the repetitive score distance optimization algorithm (algorithm 2) with initial values using the Euclidean distance algorithm.

A brief procedure of algorithm is illustrated as follows. One can set up an iterative algorithm in which, in each iteration, the values of objective function are updated such that the function is maximized. The iterations will converge to a situation in which the function value stabilized. Let the score matrix and score distance among multiple clusters be $P_{n\times 2}= \left[\widehat{p_{ik}}\ \widehat{p_{il}}\right]$, $\hat{d}_i= \widehat{p_{ik}}- \widehat{p_{il}}$, respectively. Let us define rank vector $\hat{r}_i= Rank(\widehat{p_{ik}}- \widehat{p_{il}})$, $i=1,2,\cdots,n,\ k=l=1,2,\cdots m,\ k\neq l$. The general scheme of an iterated score distance method to maximize the objective function is as follows.

Algorithm 3 Iterated Scoring Distance Optimization Algorithm

Step 1: Generate initial restriction of $M=(\max_1, \max_2,\cdots, \max_m)$
Step 2: Initial value assignment using Euclidean Distance Algorithm
Step 3: Repeat for $k=1$ to m, where m is the number of cluster do
Step 4: for $l=1$ to m do, for only $k\neq l$
Step 5: While stopping criteria not met or fixed iterations number do
Step 6: Compute the distance vector, D, for only $S_i\in\{k, l\}$
Step 7: Compute the rank vector, R of distance vector
Step 8: for $i=1$ to n do
Step 9: if $r_i\leq \max_k$ and $d_i\geq 0$ then $S_i= k$
Step 10: else if $n-r_i+1\leq \max_l$ then $S_i= l$
Step 11: else if $n-r_i+1> \max_l$ then $S_i= k$
Step 12: end for
Step 13: end while
Step 14: end for
Step 15: end for

The above scheme is the extension of algorithm 2 in Section 2.2 that enables to obtain optimal allocation for only two clusters. Although algorithm 2 provides optimal solution, it is limited to only two clusters. Based on it, we suggest a more

sophisticated one, an iterated scoring distance method with closer agreement with the other methods in linear programming.

Using algorithm 3, we assign optimal allocation as follows;

$$C = \left[\sum_{i=1}^{n} I(S_i = 1), \sum_{i=1}^{n} I(S_i = 2), \cdots, \sum_{i=1}^{n} I(S_i = m)\right] = \left[count_1, count_2, \cdots, count_m\right].$$

The advantage of the iterated score distance method is that the value of the objective function is very close to that of the other methods which provide optimal value and is tremendously fast for massive dataset.

3. Simulation and Results

The purpose of the simulations is to confirm that the proposed algorithms are efficient and effective. Two aspects that should be carefully observed are the good performance of achieving optimal solutions and the short running time to obtain the best solution in the proposed algorithm. In this simulation we compare our algorithm, called the distance method(DM) for both cases $m = 2$ and $m > 2$ with linear programming optimization problem, revised simplex method(RS), are considered with various restrictions of the number of clusters and show the simulation results of our optimal allocation scheme. We perform computations and simulations to find the values of the objective function, Q_{DM}, Q_{RS} in order to compare the following two algorithms; (a) Distance method; (b) Revised simplex method. For given restriction of the maximum number of assigned customer for each cluster to $M = [\max_1, \max_2, \cdots, \max_m]$, generate score matrix $P_{n \times m}$ from a uniform distribution with range 0 to 1. We present the numerical results obtained by the proposed algorithm and compare with values of the other algorithms. These algorithms were run 10 times to achieve average results and the standard deviation.

Simulation is to compare the distance method algorithm with revised complex method with sample sizes of $(n = 300, 1500, 3000, 4500) \times (m = 2, 3, 3, 3, 5, 5, 5)$. The performance measure used in simulation is, A, the average computed over repetitions $b = 10$ of the absolute percentage deviation of the optimal objective function value by distance method from the true optimum objective function value by revised simplex method. Table 1 displays terms of the average absolute percentage deviation defined by

$$A = \frac{1}{b} \sum_{i=1}^{b} \left| \frac{Q_{RS} - Q_{DM}}{Q_{RS}} \right| \times 100\%$$

where Q_{RS} denotes the value of the objective function of the optimal solution obtained by revised simplex method and Q_{DM} denotes the optimal value of the objective function obtained from distance method. Table 1 also indicates that the larger the sample size drawn, the closer agreement with optimal value. The resulting average absolute error rate varies in the range 0-0.1380. Also in the Table 1, the result of Case 1 shows that distance method and revised simplex method provide the exactly same optimal solutions. Thus, our algorithm actually find true optimal solutions for only two clusters ($m=2$) case. Although, the revised simplex method provides the optimal solution, it can not be used for massive dataset because of its huge computation time.

<Table 1> The average absolute percentage deviation of optimal value of objective function from the distance method to revised simplex method in SAS LP procedure with various experiments: ( ) is standard deviation

Restriction		Sample Size			
		300	1500	3000	4500
Case 1	$M=\left(\frac{n}{3}\ \frac{2n}{3}\right)$	0 (0)	0 (0)	0 (0)	0 (0)
Case 2	$M=\left(\frac{n}{3}\ \frac{n}{3}\ \frac{n}{3}\right)$	0.0202 (0.0255)	0.0015 (0.0018)	0.0009 (0.0011)	0.0005 (0.0007)
Case 3	$M=\left(\frac{2n}{6}\ \frac{3n}{6}\ \frac{3n}{6}\right)$	0.0027 (0.0053)	0.0003 (0.0008)	0.0001 (0.0003)	8.892E-6 (1.040E-5)
Case 4	$M=\left(\frac{n}{6}\ \frac{n}{6}\ \frac{4n}{6}\right)$	0.1380 (0.1026)	0.1221 (0.0405)	0.1149 (0.0229)	0.0969 (0.0240)
Case 5	$M=\left(\frac{n}{5}\ \frac{n}{5}\ \frac{n}{5}\ \frac{n}{5}\ \frac{n}{5}\right)$	0.0406 (0.0237)	0.0140 (0.0095)	0.0039 (0.0026)	0.0028 (0.0027)
Case 6	$M=\left(\frac{n}{10}\ \frac{n}{10}\ \frac{2n}{10}\ \frac{3n}{10}\ \frac{3n}{10}\right)$	0.0936 (0.0440)	0.0722 (0.0236)	0.0644 (0.0196)	0.0691 (0.0133)
Case 7	$M=\left(\frac{n}{10}\ \frac{4n}{10}\ \frac{n}{10}\ \frac{3n}{10}\ \frac{3n}{10}\right)$	0.0776 (0.0473)	0.0374 (0.0167)	0.0249 (0.0072)	0.0281 (0.0095)

Table 2 shows, for various sample size, running time and CPU time of computations for two algorithms; distance method and revised simplex method. In Table 2, X^* means that the revised simplex method in SAS fails to achieve the optimal solutions because of large dimensionality. However, the distance method can obtain the suboptimal value comparable with true optimal value. The running times to obtain the best solutions in the proposed algorithm only are in the range 1.0 - 13.0 seconds for various kinds of experiments. These results show that the solutions of distance method are close to the true solutions very quickly for all cases.

In conclusion, if n is large, which means that there are n parameters in

optimization problem, distance method is much faster than the revised simplex algorithm and more effective to assign the optimal allocation than the prioritizing method. This algorithms are implemented in SAS/IML and LP procedure in SAS/OR software solves linear programming problems by using the revised simplex method, with a Pentium processor operating at 500 MHz clock speed with 256 Mbyte RAM PC microcomputer.

<Table 2> The averaged running times and CPU times (in second) using Revised simplex method by SAS LP procedure and iterated distance method for various experiments over 10 run.

Algorithm		Sample Size			
		1500	3000	6000	9000
Revised Simple Algorithm	$M=\left(\frac{n}{3}\ \frac{2n}{3}\right)$	7	31	166	391
	$M=\left(\frac{n}{6}\ \frac{n}{6}\ \frac{4n}{6}\right)$	8	39	213	433
	$M=\left(\frac{n}{10}\ \frac{n}{10}\ \frac{2n}{10}\ \frac{3n}{10}\ \frac{3n}{10}\right)$	12	66	302	X^*
Iterated distance algorithm	$M=\left(\frac{n}{3}\ \frac{2n}{3}\right)$	1	2	3	3
	$M=\left(\frac{n}{6}\ \frac{n}{6}\ \frac{4n}{6}\right)$	1	3	5	8
	$M=\left(\frac{n}{10}\ \frac{n}{10}\ \frac{2n}{10}\ \frac{3n}{10}\ \frac{3n}{10}\right)$	2	7	11	13

4. Concluding Remarks

In this paper, we have proposed and developed efficient algorithm of linear programming in the optimal customer allocation problem for call scheduling of telephone call center. Our newly proposed algorithms tend to be significantly faster than the other algorithms such as simplex method, revised simplex method, or grid search method when we encounter the massive dataset optimization problem. We have discussed several customer allocation algorithms for massive dataset; i.e., maximum score algorithm, distance algorithm, and iterated distance algorithm. These algorithms are applicable for massive dataset and provide estimation accuracy and computational efficiency.

Our simulation results are illustrated in Table 1 and 2. We found that, without exception, the performance measures are so close or equal to being optimal that the proposed algorithms appear to offer the best combination of practicality for massive dataset and efficiency in computations. We show, through simulations, that the proposed algorithm perform almost as good as the revised simple method. Also the simulation shows that our approach can obtain the same quality with

much less computational effort.

There are, however, several issues that are currently being investigated. One such issues deals with the limitation of resources for optimal allocation. In this work, we only deal with the sufficient resources condition, $\sum_{j=1}^{m} M_i \geq n$. Another issue under current investigation is simultaneously accounting for the cost and benefit, which means that we take account for call scheduling and staffing scheduling both. This will be the subject of future manuscript.

Reference

1. Berman O. and Cutler M. (2004). Resource allocation during tests for optimally reliable software, *Computers & Operations Research*, 31, 1847-1865
2. Brown, L. D., Mandelbaum, A., Sakov, A., Shen, H., Zeltyn, S., and Zhao, L. (2002). Multifactor poisson and gamma-poisson models for call center arrival times, *Technical Report*.
3. Brown,L. ,Gans, N., Mandelbaum, A., Sakov, A., Shen, H., Zeltyn, S. and Zhao, L. (2005). Statistical Analysis of a Telephone Call Center: A Queueing-Science Perspective, *Journal of the American Statistical Association*, 100, 36-50.
4. Brown, L., and Shen, H. (2002). Analysis of service times for a bank call center data, *Technical Report, University of Pennsylvania*.
5. Dai Y. S., Xie K., Poh L. and Yang B. (2003). Optimal testing-resource allocation with genetic algorithm for modular software systems, *Journal of Systems and Software*, 66, 47-55.
6. Deslauriers, A., Pichitlamken, J., LEcuyer, P., and Avramidis, A. N. (2003). Markov chain models of a telephone call center in blend mode. *Technical report, GERAD and DIRO, University of Montreal. Preprint.*
7. Gans, N., Koole, G., and Mandelbaum, A. (2002). Telephone calls centers: a tutorial and literature review, *Technical Report*. 22
8. Gulati S. and Malcolm, S. (2001). Call center scheduling technology evaluation using simulation. *Proceedings of the 2001 Winter Simulation Conference*, 1438-1442.
9. Kaspi M. and Shabtay D. (2004). Convex resource allocation for minimizing the makes pan in a single machine with job release dates, *Computers & Operations Research*, 31, 1481-1489.
10. Koole, G., and Mandelbaum, A. (2002). Queueing models of call centers: An introduction, *Annals of Operations Research*, 113, 41-59.
11. Lee, H. (2006). Data Mining Application in Inbound Call Center, *Journal of the Korean Data and Information Science Society*, 17(2), 335-344.
12. Pichitlamken,J., Deslauriers, A., LEcuyer, P. and Avramidis, A.N. (2003).

Modeling and simulation of a telephone call center. *Proceedings of the 2003 Winter Simulation Conference*, 1805-1812.
13. Shabtay D. (2004). Single and two-resource allocation algorithms for minimizing the maximal lateness in a single machine, *Computers & Operations Research*, 31, 1303-1315.
14. Tanir, O., and Booth, R. J. (1999). Call center simulation in Bell Canada, *Proceedings of the 1999 Winter Simulation Conference*, 1640-1647.

[received date : Mar. 2007, accepted date : May. 2007]