Journal of the Society of Korea Industrial and Systems Engineering
Vol. 33, No. 1, pp.41－50, March 2010.

역방향 로지스틱스에서 용량제약을 고려한 수거망 설계문제에 관한 탐색기법

김지수* · 최현선** · 이동호*†

*한양대학교 산업공학과
**엘지 디스플레이 패널 MES팀

Search Heuristics for Capacitated Refuse Collection Network Design in Reverse Logistics

Ji-Su Kim* · Hyun-Seon Choi** · Dong-Ho Lee*†

*Department of Industrial Engineering, Hanyang University
**Panel Manufacturing Execution System Team, LG Display

본 논문에서는 수명이 다한 제품이나 소비자가 더 이상 사용하지 않는 폐기품을 다시 재사용하거나 폐기하는데 필요한 일련의 활동을 위한 역방향 로지스틱스에서의 수거망 설계 문제를 다루고 있다. 수거망 설계 문제는 수거지점의 위치와 수요지의 폐기품을 수거 지점에 할당하는 것을 결정하는 문제로 정의할 수 있으며 수거지점은 재활용품이나 폐기품이 위치한 지점 근처에 위치하고 주어진 잠재적인 위치들 중에서 결정하게 된다. 여기서, 각 수거지점은 용량제약이 있어 수거 지점에 할당되는 폐기품의 양에는 제한이 있다. 본 논문에서 다루는 수거망 설계문제에서는 수거지점을 설치하는데 필요한 고정비용과 수요지와 수거지점간의 수송비용의 합을 최소화하는 것을 목적으로 한다. 또한 대상문제를 보다 명확히 설명하기 위하여 정수계획법을 이용한 수리적 모형을 제안하였으며 문제의 복잡도 인하여 타부 서치와 시뮬레이티드 어닐링 두 가지 형태의 탐색기법을 제안하였다. 이들 탐색기법에 대하여 최대 500개의 잠재적 위치를 가지는 문제에 대하여 실험을 수행하였고 실험결과를 제시하였다.

Keywords : Reverse Logistics, Refuse Collection, Network Design, Search Heuristics

1. Introduction

There may be various recovery options when a product is at the end of its original useful life. According to Thierry et al. [27], they can be classified as repair, refurbishing, remanufacturing, cannibalization, and recycling. Another option to deal with used products is disposal, i.e., landfill and incineration. For any manufacturing firms or municipalities, the cost of disposal becomes an important variable. For example, Steinhilper [26] reported that disposal costs are 2%, 3%, 12.5% of direct production costs of laser printers, cars, and refrigerators, respectively. It becomes more so if firms are forced to implement the product take back scheme. As a result, manufacturers worldwide are increasingly facing the responsibility for their products at the end of life and must provide the methods for collection, product recovery, and even disposal [14].

논문접수일 : 2009년 07월 30일 논문수정일 : 2009년 08월 18일 게재확정일 : 2009년 08월 26일
† 교신저자 leman@hanyang.ac.kr
※ This work was supported by the Korea Science and Engineering Foundation(KOSEF) granted funded by Korea government(MEST) (Grant Code : 2009-0074736).

Under such circumstance, collection and product recovery activities give rise to an additional material flow from customers back to collectors, re-processors or producers, called the reverse logistics in the literature. In general, reverse logistics, which includes collection, recycling, remanufacturing, and reuse, can be defined as the logistics activities all the way from used products no longer required by the user to products again usable or disposable. Here, collection implies an activity that gathers used or end of life products for further treatment. Also, recycling implies material recovery without conserving any product structures, e.g., metal recycling and plastic recycling, while remanufacturing is the transformation of used or end of life products into units that satisfy exactly the same quality or other standards of new products. See Fleischmann et al. [6, 7], Dowlatshahi [4], Guide et al. [9], Ferguson and Browne [5], Lee et al. [18] for literature reviews on various aspects of reverse logistics.

This paper focuses on the refuse collection, an activity gathering used or end of life products or wastes and moving them to some points where further treatment is required. Note that refuse collection is one of important activities in reverse logistics since product recovery or even disposal cannot be carried out without collecting used products or wastes. Therefore, constructing an efficient refuse collection system is one of fundamental decision issues in reverse logistics for manufacturing firms or municipalities. Among various decisions in refuse collection, we address the problem of locating collection points as well as allocating refuses at demand points to collection points, to be called the collection network design problem in this paper. Here, the collection point can be defined as the place where recyclables or wastes near the point are gathered and moved to some points where further treatment is required. As in designing distribution networks for forward logistics, the collection network design problem considered in this paper is one of core decision problems in refuse collection systems.

Most previous research on designing reverse logistics or refuse collection networks are case studies on recycling of wastes, steel by-products, sand, carpets, automobiles, etc. Caruso et al. [2] considered a multi objective capacitated location allocation problem for waste service users, processing plants, and landfills in an Italian urban solid waste region, and suggest heuristic algorithms after formulating it as an integer programming model, and Kroon and Vrijens [11] suggested another integer linear programming model for a network design problem for returnable containers, and solve it using an existing optimal algorithm for the uncapacitated facility location model. Spengler et al. [25] considered a capacitated logistics network design problem for by-product recycling of steel in Germany, and suggested a mixed integer programming model, and Barros et al. [1] suggested heuristic algorithms for a capacitated location problem for sand recycling in Netherlands. Louwers et al. [20] performed case studies on designing carpet recycling networks in Europe and USA after suggesting an optimal algorithm for a nonlinear mathematical model. Also, Krikke et al. [12] developed an uncapacitated multi-echelon location model for designing reverse logistics networks, and suggested a mixed integer programming model and solved it using a commercial software package. They performed a case study on collecting, processing, and delivering end of life automobiles. See Realff et al. [23], Krikke et al. [13], and Lee and Dong [19] for other case studies on designing reverse logistics networks.

Unlike the above case studies, several articles deal with the theoretical aspects of the reverse logistics or collection network design problem. Jayaraman et al. [10] considered the problem of determining the numbers and locations of collection points and refurbishing facilities for designing reverse logistics networks of hazardous products, and suggested heuristic algorithms after formulating it using the two level hierarchical location model. Min et al. [22] suggested genetic algorithms for the problem of determining the numbers, locations, and sizes of collection points as well as centralized return centers after suggesting a mixed integer nonlinear programming model that maximizes the potential cost saving. Fleischmann et al. [8] considered the problem of designing forward distribution and reverse recovery networks at the same time, and suggested a mixed integer linear programming model that extends the traditional warehouse location problem. Recently, Salema et al. [24] generalized the model of Fleischmann et al. [8] by considering multiple product types with limited capacity and uncertainty in demand and return flows, and suggested a branch and bound algorithm that minimizes the sum of relevant costs. Also, Ko and Evans [17] suggested a genetic algorithm based heuristic for designing an integrated forward and return network, and Kim and Lee [16] focused on a collection network design problem and suggested several simple heuristics.

This paper considers the collection network design problem that determines the locations of collection points as well as the allocations of refuses at demand points to collection points. Here, the collection points are determined by selecting them from a given set of potential sites. Unlike the previous research on designing the whole reverse logistics networks, we narrow our scope to the refuse collection activity in order to provide

the basic model for designing collection networks. The objective is to minimize the sum of the fixed costs to open collection points and the transportation costs to move refuses at demand points to collection points. The main constraint is the capacity restriction, i.e., an upper limit on the amount of refuse allocated to a collection point.

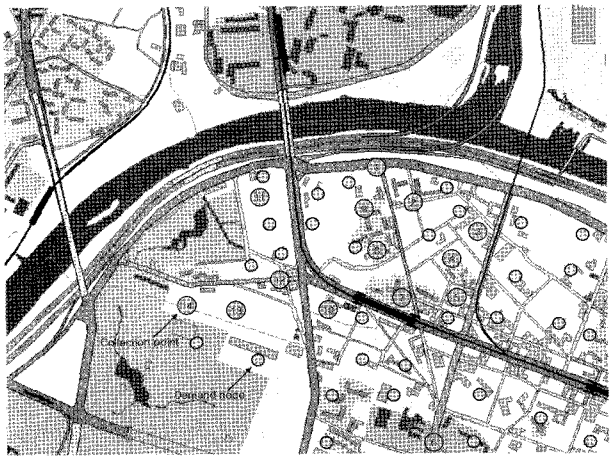
To describe the problem more clearly, we first present an integer programming model. Then, due to the complexity of the problem, two types of search heuristics, tabu search and simulated annealing, are suggested together with three neighborhood generation methods. In fact, this paper is an extension of Kim and Lee [16] (that suggest simple heuristics) in that search heuristics are suggested to find better solutions. In other words, the initial solutions for the search heuristics are obtained using the best one among those suggested by Kim and Lee [16], and their performances are shown with the amount of improvement from the initial solutions. Computational experiments were done on test problems up to 500 potential sites, and the results are reported.

The rest of this paper is organized as follows. The following section describes the problem with a mathematical formulation. Section 3 presents the search heuristics with the methods to generate neighborhood solutions, and test results are reported in Section 4, Finally, Section 5 concludes the paper with a summary and future research.

2. Problem Description

Before describing the problem, we first explain the refuse collection system. <Figure 1>, obtained from Kim et al. [15], shows a refuse collection area of a district of Seoul, Korea. (The district office operates its refuse collection system in such a way that its entire district region is decomposed into several areas.) As can be seen in the figure, the area consists of collection points and demand points. Recall that the collection point is the place where the refuses at demand points are gathered.

As stated earlier, the problem considered here has two decision variables : (a) locating collection points; and (b) allocating refuses at demand points to collection points. Here, the location decision is done by selecting them from a given set of potential sites. It is assumed that the amount of refuse at each demand point is deterministic and given in advance. The objective is to minimize the sum of fixed and transportation costs. The fixed costs are related with opening collection points, and the transportation costs, proportional to the distances as well as the refuse



<Figure 1> Refuse collection system : an example

amounts, occur when refuses are moved from demand points to collection points. It is assumed that the cost parameters are deterministic and given in advance.

The main constraint is the capacity restriction at each collection point. In other words, the amount of refuse at each demand point consumes a portion of the available capacity of the corresponding collection point, and there is an upper limit concerning the amount of refuse that can be allocated to the collection point. It is assumed that the available capacity at each collection point is given in advance. Other assumptions are : (a) each demand point is allocated to exactly one collection point, i.e., no demand splitting is allowed; (b) it is not possible to assign more than two collection points to a potential site; (c) distances between potential sites are given and symmetric; and (d) each potential site has the same fixed cost for opening a collection point.

• *Parameters*

w_i amount of refuse to be collected at potential site i, $i = 1, 2, \cdots, n$

d_{ij} distance between potential sites i and j

c_{ij} transportation cost (per refuse unit and unit distance) from potential site i to j (Potential site j is the point where a collection point is opened.)

f_j fixed cost for opening a collection point at potential site j

Q_j capacity at the collection point opened at potential site j

• *Decision variables*

y_j = 1 if potential site j is selected as a collection point, and 0 otherwise

x_{ij} = 1 if the refuse demand at potential site i is allocated

to collection point opened at potential site j, and 0 otherwise

Now, the integer programming model is given below.

• [P] Minimize

$$\sum_{j=1}^{n} f_j \cdot y_j + \sum_{i=1}^{n}\sum_{j=1}^{n} w_i \cdot d_{ij} \cdot c_{ij} \cdot x_{ij} \quad (1)$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \text{for } i = 1, 2, \cdots, n \quad (2)$$

$$x_{ij} \le y_j \qquad \text{for } i, j = 1, 2, \cdots, n \quad (3)$$

$$\sum_{i=1}^{n} w_i \cdot x_{ij} \le Q_j \cdot y_j \qquad \text{for } j = 1, 2, \cdots, n \quad (4)$$

$$x_{ij} \in \{0,1\} \qquad \text{for } i, j = 1, 2, \cdots, n \quad (5)$$

$$y_j \in \{0,1\} \qquad \text{for } j = 1, 2, \cdots, n \quad (6)$$

The objective function denotes minimizing the sum of fixed and transportation costs. Note that the transportation costs are calculated using refuse amounts as well as distances between collection and demand points. Constraint (2) ensures that each demand point be assigned to one collection point, and constraint (3) implies that demand point i can be assigned to collection point j only if there is a collection point opened at potential site j. In other words, no demand point can be assigned to a potential site without a collection point opened there. Constraint (4) denotes the capacity restriction at each collection point, i.e., the upper limit on the amount of refuse that can be assigned to the collection point. Finally, constraints (5) and (6) specify the conditions of decision variables.

Compared with the existing facility location models, the capacitated collection network design problem considered here has the characteristics of both the p-median and the hub location problems, but not exactly the same (Kim and Lee [16]). In other words, the problem [P] has the characteristic of the p-median problem in that collection points are selected among potential sites. Also, it is similar to the hub location problem since the refuses at demand points are allocated to collection points with fixed costs and capacity restrictions. However, the p-median problem does not consider fixed costs to open the locations, weights in the transportation costs, and capacity restrictions. Also, in the hub location problem, the transportation costs consist of those from the origin to hubs, between hubs, and from hubs to the destination. In fact, the hubs, which connect the origin and the destination, have only the switching or sorting function. However, the collection network design problem considered in this paper is similar to the capacitated facility location problem. Nevertheless, this paper has a certain contribution in that we defined a basic collection network design problem for reverse logistics, which can be extendable to more generalized problems.

The problem considered in this paper is NP-hard, which can be easily seen from the fact that the problem has the knapsack constraints. As reported in Kim and Lee [16], it takes too much time to solve the formulation [P] directly using a commercial software package. Due to the complexity of the problem, therefore, we suggest search heuristics that give good quality solutions in a reasonable amount of computation time.

3. Search Heuristics

This section presents the search heuristics, tabu search and simulated annealing algorithms. Before presenting the algorithms, we first explain the method to obtain an initial solution.

3.1 Obtaining an initial solution

The initial solution is obtained by the PFL-G (Preprocessing Fixed Location Generalized Assignment) algorithm suggested by Kim and Lee [16] since it gave the best result among six heuristic algorithms. More specifically, the PFL-G algorithm gave 2% gaps from the optimal solution values for small sized test instances.

Before presenting the detailed algorithm, the method to determine the number of collection points to be opened is explained first. The basic idea is to cluster the potential sites according to a non-decreasing order of transportation cost between two potential sites. The procedure is given below.

• **Procedure 1 :** (Fixing the number of collection points)

Step 1 : For all pairs of potential sites i and j, compute the transportation cost h_{ij} as follows.

$$h_{ij} = w_i \cdot d_{ij} \cdot c_{ij}$$

Then, sort all the pairs in the non-decreasing order of transportation cost, and make a list according to this order.

Step 2 : Starting from the top of the list obtained in Step 1, do the following steps. (Repeat this step until all potential sites are clustered.)

(a) Given a transportation cost h_{ij}, determine whether

two potential sites i and j can be merged without violating the collection point capacity.

(b) If such points exist, merge them.

Step 3 : Set the number of collection points to the number of clusters obtained in Step 2.

Given the number of collection points to be opened, the PFL-G algorithm obtains the initial solution by decomposing the problem into two subproblems, i.e., locating collection points from the set of potential sites and allocating demand points to the selected collection points, and then solving them iteratively, i.e., allocations are done for each alternative of the location. Here, the location is done by selecting a collection point for each cluster (obtained from the method to determine the number of collection points), and for each alternative of the set of collection points, the allocation is done by solving the generalized assignment problem (GAP) using the modified one of the MTHG algorithm of Martello and Toth [21] since it gives fast and good solutions. In general, the GAP is the problem of assigning multiple tasks to agents subject to the availability of a single resource type consumed by the agents in performing these tasks. As noted in Kim and Lee [16], in our application, each demand point can be represented by a task, and each collection point can be represented by an agent.

The procedure for the modified MTHG algorithm is given as follows.

- **Procedure 2 :** (Modified MTHG algorithm for allocation)

Step 1 : Initialization

(a) For all unassigned demand points, find the one (denoted by point i) having the maximum difference between the largest and the second largest values of g_{ij}, where $g_{ij} = w_i \cdot d_{ij}$ for all i and j. (Recall that w_i and d_{ij} denote the amount of refuse at potential site i and the distance between potential sites i and j, respectively.)

(b) Allocate demand point i to collection point j for which the desirability measure g_{ij} has the minimum value without violating the capacity of the collection point. If collection point j cannot be allocated, do this step for the collection point that has the second minimum value of the desirability measure, and so on.

(c) If all demand points are allocated, save the initial solution and go to Step 2. Otherwise, go to step (a).

Step 2 : Improvement

(a) For each demand point i, find the set A_i of collection points that give improvements without violating their capacity restrictions. Here, the set A_i can be described as follows (In the initial solution, let demand point i be allocated to collection point j^*, i.e., $x_{ij}^* = 1$).

$A_i = \{j \neq j^* \mid g_{ij} < g_{ij}^* \text{ and } w_i \leq R_j\}$,

where R_j denotes the remaining capacity of collection point j. If $A_i = \phi$ for all i, i.e., no improvement can be made, stop the algorithm. Otherwise, go to Step (b).

(b) Perform the reallocation that gives the maximum amount of improvement after evaluating all possible candidates. Here, a reallocation implies that demand point i is assigned to the set A_i of collection points. Repeat this step until there is no improvement.

3.2 Algorithms

3.2.1 Tabu Search Algorithms

Tabu search (TS) is one of the well known local search techniques that have been successfully applied to various combinatorial optimization problems. Starting from an initial solution (obtained by the method described earlier), TS generates a new alternative S' in the neighborhood of the original alternative S. This is usually called a move, which can be made to a neighborhood solution even though it is worse than the given solution. This makes a TS algorithm escape from a local optimum in its search for the global optimum. In order to avoid cycling, TS defines a set of tabu moves (forbidden), and these moves are stored in a set Λ, called the tabu list. Elements of Λ define all tabu moves that cannot be applied to the current solution. The size of Λ is bounded by parameter l, called the tabu list size. If $|\Lambda| = 1$, before adding a move to Λ, the oldest element in it is removed. Note that a tabu move can be allowed if it creates a solution better than the best objective value obtained so far, called the aspiration criterion in the literature.

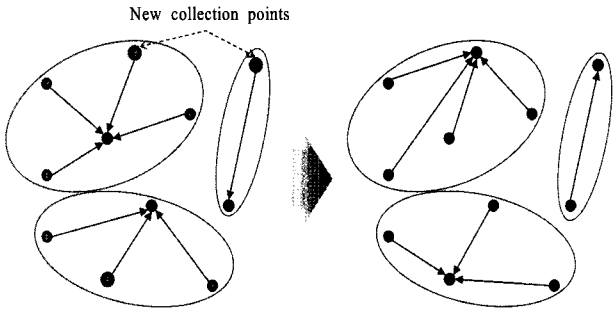
The TS algorithms are explained with : (a) solution representation method; (b) neighborhood generation methods, i.e., set of possible moves applicable to the current solution; (c) definition of tabu moves with the tabu list size; and (d) termination condition.

The solution, location and allocation, is represented as two vectors $U = (u_1, u_2, \cdots, u_n)$ and $V = (v_1, v_2, \cdots, v_n\}$, where ui denotes the index of the collection point assigned to potential site i and $v_i = 1$ if potential site i is selected as a collection point, and 0 otherwise. Note that the objective value of a given solution can be easily calculated while checking its feasibility.

In this paper, we suggest three neighborhood generation methods, called re-clustering, adding, and removal and insertion. (Therefore, we suggest three TS algorithms that are different in the neighborhood generation method.) Each of the neighborhood generation methods is explained below.

• *Re-clustering method*

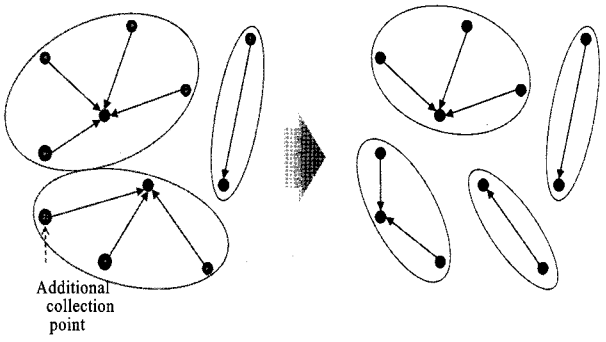
In this method, a collection point is removed from a randomly selected cluster and a new collection point is selected from a set of potential sites (except for the removed collection point) within the cluster. Then, the demand points within the selected cluster and the removed collection point are reallocated to the new collection point. Hence, the number of collection points before and after generating a neighborhood solution remains the same. See <Figure 2> for an example of the re-clustering method.



<Figure 2> Neighborhoods : re-clustering

• *Adding method*

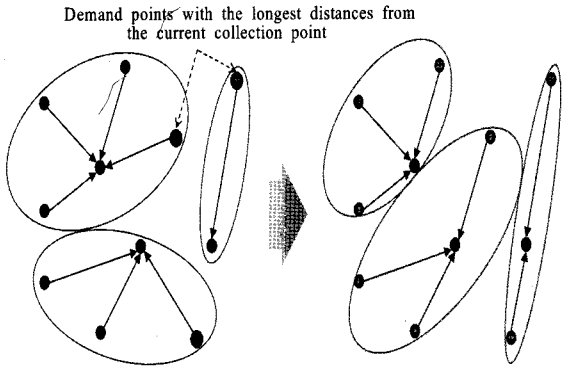
This method generates a neighborhood solution by opening an additional collection point randomly and allocating the demand points nearest to the newly opened collection point while considering its capacity restriction. Hence, the number of collection points is increased by one, and also there are certain changes in the allocations. See <Figure 3> for an example.



<Figure 3> Neighborhoods : adding

• *Removal and insertion method*

In this method, the demand point with longest distance from the corresponding collection point is removed for each cluster. Then, each removed demand point is reallocated to the collection point that has the minimum distance while considering the capacity restriction. See <Figure 4> for an example.



<Figure 4> Neighborhoods : Removal and Insertion

Tabu moves are maintained by storing the solutions that have been visited. In other words, during the search process two lists, locations of collection points and allocations of demand points to collection points, were maintained and checked for tabu moves. As an exceptional case, a tabu move can be allowed to be chosen if it generates a solution better than the incumbent solution, i.e., the best objective value obtained so far. Finally, the TS algorithms were stopped if no improvements have been made for a certain number of consecutive iterations, denoted by LTS in this paper.

3.2.2 Simulated Annealing Algorithms

Simulated annealing (SA) also attempts to move from the current solution to one of its neighborhoods. Starting from an initial solution, SA generates a new solution S' in the neighborhood of the original S. Then, the change in the objective value, $\Delta = C(S')-C(S)$, is calculated, where $C(\bullet)$ is the objective value of solution $\bullet$. If $\Delta < 0$ (for a minimization problem), the transition to the new solution is accepted. Otherwise, the transition to the new solution is accepted with a specified probability denoted by the function $exp(-\Delta/t)$, where t is a control parameter called the temperature. By allowing the moves that increase the objective value, the SA can escape from a local minimum. In this paper, three SA algorithms are suggested according to the three neighborhood generation methods explained earlier. In fact, the SA algorithms are same as the TS algorithms

except for the basic search mechanism.

There are four generic parameters that characterize SA algorithms : (a) initial temperature (t_0); (b) epoch length (α), i.e., number of transitions made with the same temperature; (c) rule specifying how the temperature is reduced; and (d) termination condition. In this paper, the temperature is decreased using the commonly used equation, $t_k - r \cdot t_{k-1}$, where t_k is the temperature used during the kth epoch and r is a positive constant with a value less than 1, called the cooling ratio in the literature. Finally, the algorithms were terminated when there is no improvement for a certain number of iterations, denoted by L_{SA}.

4. Computational Results

To show the performances of the six search heuristics, computational experiments were done on a number of test problems, and the results are reported in this section. The six search heuristics are compared with the PFL-G algorithm, the best existing heuristic among those suggested by Kim and Lee [16]. In fact, the PFL-G algorithm gave near optimal solutions for the small-size test instances, i.e., 2% gap in average from the optimal solution values. The performance measures used are : (a) percentage improvements over the PFL-G algorithm; and (b) CPU

<Table 1> Test Results for Search Heuristics : Percentage Improvements

(a) Problems with loose capacity

Number of potential sites	TS reclustering	TS adding	TS removal/ insertion	SA reclustering	SA adding	SA removal/ insertion
10	0.00(0.0)*	0.00(0.1)	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)
20	0.21(0.1)	0.08(0.1)	0.08(0.1)	0.21(0.3)	0.21(0.4)	0.08(0.1)
30	0.52(0.1)	0.30(0.3)	0.30(0.2)	0.32(0.3)	0.32(0.3)	0.30(0.0)
40	1.14(0.1)	0.55(0.0)	0.55(0.0)	0.98(0.1)	0.98(0.1)	0.55(0.1)
50	0.23(0.2)	0.18(0.3)	0.18(0.1)	0.23(0.1)	0.23(0.3)	0.18(0.2)
100	0.21(0.1)	0.10(0.0)	0.10(0.1)	0.19(0.1)	0.19(0.0)	0.10(0.1)
200	0.21(0.1)	0.17(0.0)	0.18(0.0)	0.27(0.3)	0.24(0.0)	0.17(0.1)
300	0.15(0.1)	0.13(0.0)	0.13(0.0)	0.17(0.0)	0.17(0.0)	0.14(0.0)
400	0.13(0.0)	0.12(0.1)	0.12(0.2)	0.12(0.0)	0.13(0.1)	0.12(0.1)
500	0.06(0.0)	0.06(0.1)	0.06(0.1)	0.06(0.0)	0.07(0.1)	0.06(0.2)
Average	0.29(0.1)	0.17(0.1)	0.17(0.1)	0.26(0.1)	0.25(0.1)	0.17(0.1)

Note : * average of the percentage improvements over initial solutions out of 10 problems and standard deviation(in parenthesis).

(b) Problems with medium capacity

Number of potential sites	TS reclustering	TS adding	TS removal/ insertion	SA reclustering	SA adding	SA removal/ insertion
10	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)
20	1.20(0.2)	2.35(0.1)	2.24(0.5)	1.22(0.2)	1.27(0.1)	2.26(0.4)
30	1.71(0.2)	1.94(0.2)	1.96(0.1)	1.57(0.1)	1.52(0.2)	1.80(0.2)
40	2.06(0.3)	2.05(0.1)	1.94(0.4)	1.84(0.3)	1.99(0.2)	1.86(0.3)
50	1.59(0.2)	2.76(0.2)	2.85(0.2)	1.82(0.1)	2.02(0.1)	2.94(0.1)
100	1.47(0.3)	2.61(0.2)	2.55(0.2)	1.96(0.2)	1.75(0.2)	2.48(0.2)
200	1.80(0.1)	3.01(0.2)	2.91(0.3)	1.98(02)	1.89(0.2)	2.80(0.2)
300	1.63(0.3)	3.33(0.3)	3.29(0.1)	1.87(0.2)	2.10(0.2)	3.21(0.2)
400	1.52(0.2)	3.44(0.1)	3.47(0.2)	2.05(0.2)	1.86(0.1)	3.42(0.2)
500	1.48(0.2)	3.26(0.2)	3.25(0.1)	2.40(0.2)	2.01(0.1)	3.18(0.2)
Average	1.44(0.2)	2.47(0.2)	2.44(0.2)	1.67(0.2)	1.64(0.1)	2.39(0.2)

(c) Problems with tight capacity

Number of potential sites	TS reclustering	TS adding	TS removal/ insertion	SA reclustering	SA adding	SA removal/ insertion
10	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)	0.00(0.0)
20	1.59(0.4)	3.62(0.5)	3.39(0.5)	1.22(0.2)	1.33(0.7)	3.64(0.9)
30	2.13(0.4)	2.58(0.7)	2.52(0.7)	1.81(0.3)	1.71(0.3)	2.50(0.3)
40	2.37(0.3)	2.46(0.6)	2.32(0.5)	1.69(0.5)	1.83(0.8)	2.36(0.5)
50	2.34(0.7)	4.34(0.7)	4.51(0.1)	2.38(0.1)	2.80(0.2)	4.72(0.4)
100	2.12(0.3)	4.12(0.8)	4.00(0.1)	2.73(0.3)	2.30(0.4)	4.06(0.2)
200	2.62(0.1)	4.74(0.3)	4.64(0.7)	2.67(0.3)	2.53(0.8)	4.55(0.4)
300	2.51(0.3)	5.53(0.4)	5.45(0.9)	2.57(0.2)	2.94(0.7)	5.48(0.3)
400	2.31(0.1)	5.76(0.6)	5.82(0.9)	2.97(0.2)	2.59(0.6)	5.77(0.4)
500	2.30(0.1)	5.45(0.5)	5.43(0.4)	3.73(0.3)	2.94(0.6)	5.50(0.9)
Average	2.03(0.3)	3.86(0.5)	3.81(0.5)	2.18(0.2)	2.10(0.5)	3.86(0.4)

seconds. Here, the percentage improvement of search heuristic a for a test problem is measured as

$$100 \cdot (C_I - C_a)/C_I,$$

where C_I and C_a are the objective values of the initial solution obtained by the PFL-G algorithm and search heuristic a, respectively. All the algorithms were coded in C and the test was done on a workstation with a Xeon processor operating at 3.2 GHz speed.

To find the appropriate values for the parameters of the search heuristics, preliminary experiments were done on representative test problems. For the TS algorithms, several values for tabu list size l and L_{TS} for termination condition were tested, and they were set to 50 and 200000, respectively. Also, the parameters of the SA algorithms were set as (t_0, a, r, L_{SA}) = (1, 20, 0.997, 200000), where t_0, a, r and L_{SA} represent the initial temperature, the epoch length, the cooling ratio, and the parameter for the termination condition, respectively.

For the test, 300 problems were generated randomly, i.e. 10 problems for each of 30 combinations of ten levels for the number of potential sites (10, 20, 30, 40, 50, 100, 200, 300, 400, and 500) and three levels for the tightness of the collection point capacity (tight, medium, and loose). The problem data were generated using the method of Daskin [3]. More specifically, the x and y coordinates of potential sites were generated from $DU(0,$

<Table 2> Test Results for Search Heuristics : CPU Seconds

Number of potential sites	TS reclustering	TS adding	TS removal/ insertion	SA reclustering	SA adding	SA removal/ insertion
10	1.0*	0.9	1.0	0.9	1.2	1.1
20	1.0	0.8	0.9	1.1	1.1	1.1
30	1.9	1.5	1.5	2.1	2.0	1.7
40	3.3	2.5	2.7	4.2	3.5	2.9
50	4.6	2.8	3.1	5.9	5.1	3.2
100	22.9	7.6	10.5	29.3	19.7	10.5
200	139.6	39.6	44.8	216.6	199.0	38.2
300	305.4	97.5	97.3	505.5	264.9	86.6
400	455.1	200.5	196.5	803.1	339.7	178.9
500	540.8	366.5	357.6	1143.1	466.2	307.9
Average	147.6	72.0	71.6	271.2	130.3	63.2

Note : * average CPU second out of 10 problems.

100), i.e., a square shaped grid with a size of 100 × 100 unit lengths, where $DU(l, u)$ denotes the discrete uniform distribution with range $[l, u]$. The distance between two potential sites was obtained by calculating its Euclidean distance and then rounding it to the nearest integer, and refuse demands at potential sites were generated from $DU(10, 40)$. The fixed costs were generated from $DU(12000, 15000)$, and the unit transportation cost was set to 10. Finally, the capacity of each collection point was set to 40, 80, and 120 for tight, medium, and loose capacity cases, respectively.

Test results are given in <Table 1> that summarizes the average percentage improvements over the initial solutions obtained by the PFL-G algorithm. It can be seen from the table that the search heuristics suggested in this paper improve the initial solutions, especially as the capacity restriction gets tighter. Although there was not much difference between the TS and the SA algorithms, the TS algorithms were slightly better than the SA algorithms in overall average. Among the three neighborhood generation methods, the adding and the removal and insertion methods were better than the reclustering method (except for the case of loose capacity restriction) since they can search larger solution spaces. <Table 2> shows the CPU seconds of the search heuristics. As expected, the search heuristics required much longer CPU seconds than the PFL-G algorithm. However, all the heuristics gave the solutions within 8 minutes in average, which implies that our search heuristics can be used for practical applications. From the computational test, in summary, we recommend the TS algorithm with the adding method or the removal and insertion method with respect to solution quality and computation time.

5. Concluding Remarks

We considered a network design problem for collecting refuses in reverse logistics. The problem, called the capacitated collection network design problem, is to determine the location of collection points as well as the allocation of refuses at demand points to collection points while satisfying the capacity constraint at each collection point. The objective is to minimize the sum of the fixed costs to open collection points and the transportation costs between demand and collection points. Two types of search heuristics, tabu search and simulated annealing, were suggested that incorporate the methods to generate neighborhood solutions. From the computational experiments on test problems up to 500 potential sites, we showed that both search heuristics gave improvements over an existing algorithm.

As one of the early research on designing collection networks for reverse logistics, this research can be extended in several directions. First, the optimal solution algorithms are worth to be developed in the theoretical aspect. Second, it is required to extend the basic model into the ones with other features of refuse collection systems such as stochastic amounts of refuses, multiple refuse types, etc. Finally, it is needed to perform certain case studies on designing real refuse collection systems.

References

[1] Barros, A. I., Dekker, R., and Scholten, V.; "A two level network for recycling sand : a case study," *European Journal of Operational Research*, 110 : 199-214, 1998.

[2] Caruso, C., Colorni, A., and Paruccini, M.; "The regional urban solid waste management system : a modelling approach," *European Journal of Operational Research*, 70 : 16-30, 1993.

[3] Daskin, M. S.; "GENRAND2 : A random network generator," Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, USA, 1993.

[4] Dowlatshahi, S.; "Developing a theory of reverse logistics," Interfaces, 30 : 143-155, 2000.

[5] Ferguson, N. and Browne, J.; "Issues in end-of-life product recovery and reverse logistics," *Production Planning and Control*, 12 : 534-547, 2001.

[6] Fleischmann, M., Bloemhof Ruwaard, J. M., Dekker, R., Van der Laan, E., Van Nunen, J. A. E. E., and Van Wassenhove, L. N.; "Quantitative models for reverse logistics : a review," *European Journal of Operational Research*, 103 : 1-17, 1997.

[7] Fleischmann, M., Krikke, H. R., Dekker, R., and Flapper, S. D. P.; "A characterization of logistics networks for product recovery," Omega, 28 : 653-666, 2000.

[8] Fleischmann, M., Beullens, P., Bloemhof Ruwaard, J. M., and Van Wassenhove, L. N.; "The impact of product recovery on logistics network design," *Production and Operations Management*, 10 : 156-173, 2001.

[9] Guide, Jr. V. D. R., Jayaraman, V., Srivastava, R., and Benton, W. C.; "Supply-chain management for recoverable manufacturing systems," Interfaces, 30 : 125-142, 2000.

[10] Jayaraman, V., Patterson, R. A., and Rolland, E.; "The design of reverse distribution networks : models and solution procedures," *European Journal of Operational Research*,

150 : 128-149, 2003.

[11] Kroon, L. and Vrijens, G.; "Returnable containers : an example of reverse logistics," *International Journal of Physical Distribution and Logistics Management*, 25 : 56-68, 1995.

[12] Krikke, H. R., Kooi, E. J., and Schuur, P. C.; "Network design in reverse logistics : a quantitative model," *Lecture Notes in Economics and Mathematical Systems*, 480 : 45-62, 1999a.

[13] Krikke, H. R., van Harten. A., and Schuur, P. C.; "Business case océ : reverse logistics network re-design for copiers," OR Spectrum, 21 : 381-409, 1999b.

[14] Klausner, M. and Hendrickson, C. T.; "Reverse logistics strategy for product take back," Interfaces, 30 : 156-165, 2000.

[15] Kim, J.-D., Choi, H.-S., and Lee, D.-H.; "A case study on the stochastic vehicle routing in a refuse collection system," *Journal of the Korean Society of Supply Chain Management*, 7 : 57-65, 2007.

[16] Kim, J.-S. and Lee, D.-H.; "Heuristics algorithms for capacitated collection network design in reverse logistics," *International Journal of Management Science*, 14 : 45-66, 2008.

[17] Ko, H. J. and Evans, G. W.; "A genetic algorithm based heuristic for the dynamic integrated forward/reverse logistics network for 3PLs," *Computers and Operations Research*, 34 : 346-366, 2007.

[18] Lee, D.-H., Kim, H.-J., and Kim, J.-S.; "Reverse logistics : research issues and literature review," *Journal of the Korean Institute of Industrial Engineers*, 34 : 270-288, 2008.

[10] Lee. D. H. and Dong. M.; "A heuristic approach to logistics network design for end of lease computer products recovery," Transportation Research Part E : *Logistics and Transportation Review*, 44 : 455-474, 2008.

[19] Louwers, D., Kip, B. J., Peters, E., Souren, F., and Flapper, S. D. P.; "A facility location allocation model for reusing carpet materials," *Computers and Industrial Engineering*, 36 : 855-869, 1999.

[20] Martello, S. and Toth, P.; Knapsack Problems : Algorithms and Computer Implementations, John Wiley and Sons, 1990.

[21] Min, H., Ko, H. J., and Ko, C. S.; "A genetic algorithm approach to developing the multi echelon reverse logistics network for product returns," Omega, 34 : 56-69, 2006.

[22] Realff, M. J., Ammons, J. C., and Newton, D.; "Carpet recycling : determining the reverse logistics system design," *Journal of Polymer Plastics Technology and Engineering*, 38 : 547-567, 1999.

[23] Salema, M. I. G., Barbosa Povoa, A. P., and Novais, A. Q.; "An optimization model for the design of a capacitated multi product reverse logistics network with uncertainty," *European Journal of Operational Research*, 179 : 1063-1077, 2007.

[24] Spengler, Th., Püchert, H., and Rentz, P. O.; "Environmental integrated production and recycling management," *European Journal of Operational Research*, 97 : 308-326, 1997.

[25] Steinhilper, R.; Remanufacturing : the Ultimate Form of Recycling, Fraunhofer IRB Verlag, 1998.

[26] Thierry, M., Salomon, M., van Nunen, J., and van Wassenhove, L.; "Strategic issues in product recovery management," *California Management Review*, 37 : 114-135, 1995.