Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computers & Operations Research 33 (2006) 3624–3642
computers & operations research
www.elsevier.com/locate/cor

Waste collection vehicle routing problem with time windows

Byung-In Kim[a,*], Seongbae Kim[b], Surya Sahoo[b]

[a] *Department of Industrial and Management Engineering, POSTECH, Pohang, Kyungbuk, 790-784, Korea*
[b] *Institute of Information Technology, Inc., 2204 Timberloch Place Suite 225, The Woodlands, TX 77380, USA*

Available online 21 November 2005

Abstract

In this paper, we address a real life waste collection vehicle routing problem with time windows (VRPTW) with consideration of multiple disposal trips and drivers' lunch breaks. Solomon's well-known insertion algorithm is extended for the problem. While minimizing the number of vehicles and total traveling time is the major objective of vehicle routing problems in the literature, here we also consider the route compactness and workload balancing of a solution since they are very important aspects in practical applications. In order to improve the route compactness and workload balancing, a capacitated clustering-based waste collection VRPTW algorithm is developed. The proposed algorithms have been successfully implemented and deployed for the real life waste collection problems at Waste Management, Inc. A set of waste collection VRPTW benchmark problems is also presented in this paper.

Waste collection problems are frequently considered as arc routing problems without time windows. However, that point of view can be applied only to residential waste collection problems. In the waste collection industry, there are three major areas: commercial waste collection, residential waste collection and roll-on-roll-off. In this paper, we mainly focus on the commercial waste collection problem. The problem can be characterized as a variant of VRPTW since commercial waste collection stops may have time windows. The major variation from a standard VRPTW is due to disposal operations and driver's lunch break. When a vehicle is full, it needs to go to one of the disposal facilities (landfill or transfer station). Each vehicle can, and typically does, make multiple disposal trips per day. The purpose of this paper is to introduce the waste collection VRPTW, benchmark problem sets, and a solution approach for the problem. The proposed algorithms have been successfully implemented and deployed for the real life waste collection problems of Waste Management, the leading provider of comprehensive waste management services in North America with nearly 26,000 collection and transfer vehicles.

Keywords: Waste collection; Vehicle routing problems with time windows; Capacitated clustering

* Corresponding author. Tel.: +82 54 279 2371; fax: +82 54 279 2870.
E-mail address: bkim@postech.ac.kr (B.-I. Kim).

0305-0548/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2005.02.045

1. Introduction

We have developed and implemented various vehicle routing algorithms for Waste Management, Inc. (WM), the leading provider of comprehensive waste management services in North America with nearly 26,000 collection and transfer vehicles (see [1]). In this paper, we present the algorithms for waste collection vehicle routing problem with time windows (VRPTW) with consideration of multiple dump site locations and drivers' lunch breaks.

The waste collection business is divided into three major areas: commercial, residential and roll-on-roll-off (see [2]). Each area includes municipal solid waste and recycling material, and each is very different from the others. The commercial waste collection involves servicing customers such as strip malls, restaurants and small office buildings. Each commercial route of WM may service 60–400 customers, with two or three disposal trips to dump sites each day. Depending upon the customer base, the same driver may visit the same customer multiple times in one week. The weekly service schedule is fairly static, as most customers do not change the frequency of service often. The commercial customers may have time windows.

The residential waste collection generally involves servicing private homes. The number of homes a residential route may service varies widely from 150 to 1300 homes every day. The frequency of service per week will vary based on the climate, geography, competition and price of service. In the northern states, it is common to service a residential home once per week. Conversely, in the southern states, it is more typical to service a residential home two times per week. Once the weekly frequency is determined for a set of routes, the schedule repeats itself every week. The service activities for Monday will repeat every non-holiday Monday.

The roll-on-roll-off collection introduces a different routing problem. The differentiator between roll-on-roll-off and commercial is the size of the container. A typical commercial container is eight loose yards, while a roll-on-roll-off container may range from 20 to 40 loose yards and only one container may be serviced at a time. Note that 1 cubic yard is $0.765\,m^3$. While hauling these large containers, it is common for each container to be disposed of and returned to the original customer's location. To complicate the problem, many different approaches are used to make this operation more efficient. One example of servicing the customer would be to first deliver an additional empty container at the customer location, pick up the full container, travel to a disposal facility and then dispose of the contents. At this point, the vehicle may service another customer with the same size container. The difficulty arises when a driver is scheduled to perform different types of services throughout the day to customers with different container sizes and different service requirements. Driver-experience level, vehicle types, container types, material types and security clearance are all contributing factors when creating industrial routes.

In this paper, we mainly focus on the daily commercial waste collection problem. The problem can be characterized as a variation of vehicle routing problems with time windows and intermediate facilities (VRPTW–IF). The weekly service schedule is assumed predetermined. The algorithm for the weekly service schedule will be published in other literature in the near future.

It is hard to find research work that considers multiple disposal trips and drivers' lunch break in the VRPTW literature. In most of the literature, each vehicle is assumed to depart from the depot, serve (pick-up or delivery) only one route consisting of multiple stops, and return to the depot as shown in Fig. 1. Considering disposal trips is not a simple matter. One may attempt to solve the problem by generating initial route without considering disposal trips and then inserting disposal trips to the route. However, that method does not work because the route feasibility is not guaranteed in the disposal trip

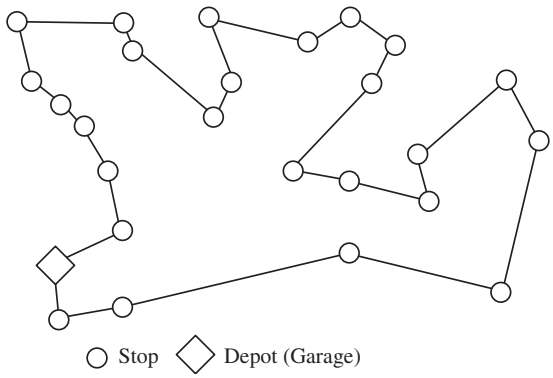

Fig. 1. A route sequence of one vehicle without considering dump operation.

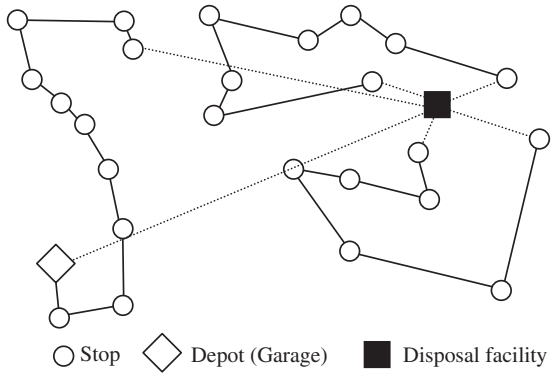

Fig. 2. A route sequence of one vehicle considering dump operation with one dump site.

insertion stage. The arrival time of a vehicle at a stop following a disposal trip will be changed and the time may not be in the stop's allowable time window. Efficiency of the stop sequence of the approach is also problematic. As shown in Figs. 1 and 2, the optimal stop sequence of the route without consideration of disposal trips is quite different from the sequence of the route generated with the consideration. When there are more than one disposal facilities, the problem is more complicated since we need to determine which facility to use in addition to when to use it. Fig. 3 shows an example case that has three facilities.

Drivers' lunch break is not usually considered either in the literature. One may also attempt to first solve the problem without considering the break and insert the break afterward. However, it does not work for the same reason. Fig. 4 illustrates an example. If we get a feasible sequence without considering the break such as Fig. 4(a), a possible position of the lunch break is between stop 2 and 3. Insertion of the break between stop 2 and 3 makes the sequence infeasible as shown in Fig. 4(b). The start time of stop 5, 15:10, is not within its allowable start time window, which is [13:00–15:00]. Fig. 4(c) shows a feasible solution of the case. In order to explicitly consider disposal trips and the lunch break, we extend the well-known Solomon's insertion algorithm.

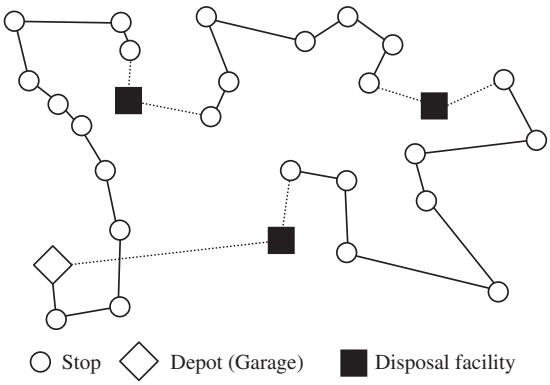

Fig. 3. A route sequence of one vehicle considering dump operation with multiple dump sites.

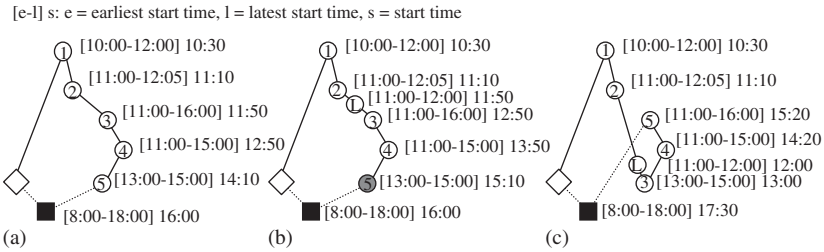

Fig. 4. Lunch break consideration of one vehicle.

While minimizing the number of vehicles and total traveling time is the major objective of vehicle routing problems in the literature, here we also consider the route compactness of a solution since it is an important criterion in many practical applications. A solution with better route compactness has smaller number of crossovers among the routes. In order to improve the route compactness, we develop a clustering-based waste collection VRPTW algorithm. After the number of vehicles required to serve the given stops is estimated, the stops are clustered based on their locations. At this stage, clusters are formed such that each cluster can be served by a single vehicle. Each cluster is solved by the extended insertion algorithm developed in this research. The route compactness of the solution can be improved further by moving stops among clusters. When there is no feasible solution with the estimated number of vehicles, the procedure is repeated with an increased number of vehicles. Computational results show the effectiveness of the proposed approach.

This paper is organized as follows. After we briefly review the related literature on VRPTW and waste collection problem in Section 2, detailed description of the problem is followed in Section 3. The extended insertion algorithm and a clustering-based waste collection VRPTW algorithm are presented in Sections 4 and 5, respectively. Section 6 shows the experimental results and benchmark problems. Section 7 provides the concluding remarks.

2. Literature review

In this section, we briefly review the literature on VRPTW, especially heuristic methods, and waste collection. Solomon [3] proposes a benchmark problem set for the vehicle routing problem with time windows and conducts a computational study of several heuristic algorithms using the set. Among the tested heuristics, an insertion heuristic, which is known as Solomon's insertion heuristic, generates the best routing schedules in many cases. While Solomon's insertion heuristic builds a route one at a time in a serial manner, Potvin and Rousseau's [4] method builds routes in parallel. They use the Solomon's heuristic to find the initial seed customers and number of vehicles. A regret measure is used to select a customer to be inserted at each iteration. If there is a feasible solution with the fixed number of vehicles, they reduce the number of vehicles and check the feasibility. Otherwise, the algorithm searches for a feasible solution with an increased number of vehicles.

Rochat and Taillard [5] present a probabilistic tabu search method for vehicle routing problems. In their method, multiple initial solutions are generated by a local search heuristic. Each route in a generated solution is labeled with the objective value of the solution (e.g., travel time) and is added to a route pool called adaptive memory. Depending on the label of a route, the route has a probability to be chosen. A route from a good solution is assigned a higher probability. Then a new solution is constructed by choosing routes probabilistically from the pool. In each selection, routes in the pool that have customers belonging to the routes already chosen are ignored. After this, if there are remaining customers to be inserted and there are no more routes to be selected, the customers are inserted by an insertion heuristic. The new solution is labeled with its objective value and added to the route pool. The steps are repeated until a stopping criterion is satisfied.

Taillard et al. [6] use a new edge exchange heuristic named CROSS and tabu search to improve the vehicle routing solutions. They restrict exchanging of customers between any two routes to subsets of consecutive customers. Constant time methods of evaluation and a feasibility test for an exchanged solution using an approximation function are also presented. To speed up computation, a decomposition and reconstruction approach is used. Taillard et al. [7] consider a vehicle routing problem with multiple use of vehicles in which multiple routes can be served by the same vehicle during a planning period and present a tabu search heuristic for the problem. The time windows of the stops and intermediate facilities such as landfills are not considered in their discussion.

Weigel and Cao [8] present a case study of application of VRPTW algorithms for Sears home delivery problem and technician dispatching problem. They follow a cluster-first-route-second method and discuss three main routines: origin-destination (OD) matrix construction, route assignment, and route improvement routines. They apply a shortest-path algorithm to a geographic information system (GIS) to obtain OD matrix, i.e., travel time between any two stops. For the route assignment routine (clustering), an algorithm called multiple-insertion, which is similar to the parallel insertion algorithm of Potvin and Rousseau [4] is developed. As an objective function, the weighted combination of travel time, wait time, and time window violation is used. They propose an intra-route improvement algorithm and a neighborhood inter-route improvement algorithm that improves the solution quality by transferring and exchanging stops between two routes. In order to enhance the improvement performance, tabu search is applied to the improvement algorithms.

Tung and Pinnoi [9] modify Solomon's insertion algorithm and apply it to a waste collection problem in Hanoi, Vietnam. In addition to the considerations of the standard VRPTW, they consider a landfill operation that is the dumping of the collected garbage at the landfill, and inter-arrival time constraints

between two consecutive visits at a stop. They incorporate the landfill operation by assuming that a vehicle starts a new route from the depot after landfill. Or-opt and 2-opt algorithms are adopted to improve the solution quality. The problem considered in this paper is similar to the waste collection problem of Tung and Ponnoi [9] but there are also distinctive characteristics between the two. There is only one landfill location in theirs, whereas there are more than one landfill locations in ours. Multiple landfill locations make the problem more complicated. While there are three distinctive shifts in their problem so that they can solve it separately, there is only one shift in our problem. While they assume that the drivers take a lunch break between shifts so that the break does not need to be considered, the drivers take a 1-h lunch break between 11 a.m. and 1 p.m. in our problem so the lunch break needs to be considered explicitly.

Poot et al. [10] present several non-traditional constraints of real-life vehicle routing problems such as multiple capacity constraints, vehicle type constraints, region constraints, served first or last constraint, and multiple time windows. They propose a savings-based method for the problems that have those constraints.

Angelelli and Speranza [11] address the periodic vehicle routing problem with intermediate facilities (PVRP–IF). When a vehicle visits an intermediate facility, its capacity will be renewed. They propose a tabu search algorithm with four move operators: move a customer in the same day, change the visiting schedule, redistribution of customers, and simplification of intersection. Initial solutions are built by choosing a visiting schedule randomly to each customer and constructing vehicle routes on each day using iterative insertion procedure. Angelelli and Speranza [12] applied their algorithm for estimating the operating costs of different waste-collection systems: traditional system with three-man crew, side-loader system, and side-loader system with demountable body. The differentiator between their problem and ours is the time windows of the stops and the facilities. Our problem requires explicit consideration of time windows.

Teixeira et al. [13] apply a heuristic approach for a PVRP for the separate collection of three types of waste: glass, paper, and plastic/metal. The approach has three phases: define a zone for each vehicle, define waste type to collect on each day, and select the sites to visit and sequence them. Eisenstein and Iyer [14] use a Markov decision process to model the residential waste collection problem in the city of Chicago. They model the weight and time required to collect waste from a city block as normally distributed random variables. Action in their Markov decision process is the choice of route that visits the dumpsite once or twice. Chang et al. [15] discuss how combining GIS functions with analytical models can help analyzing alternative solid waste collection strategies for a metropolitan city in Taiwan. Mourao and Almeida [16] model the residential garbage collection problem in a quarter of Lisbon, Portugal as a capacitated arc routing problem, and propose two lower-bounding methods and a route-first, cluster-second heuristic method.

3. The waste collection vehicle routing problem with time windows

As with a typical VRPTW, at WM, it is assumed that there is a single depot per site, an operational area (such as a city or county), a finite number of homogeneous vehicles, a set of stops, and dump sites (transfer stations or landfill facilities). The constraints specific to landfills are a major component of the routing model for waste collection companies. When a vehicle is full, it needs to go to the closest available disposal facility. Each vehicle can, and typically does, make multiple disposal trips per day.

Each stop has a time window, i.e., earliest and latest allowable service starting time, and its pick-up amount or demand. Each vehicle has its capacity and time window, i.e., ready time and due depot return time. Vehicles start from the depot, pick-up materials from the stops until they are full, dump them at one of the dump sites, repeat pick-up and dumping, and finally return to the depot. The vehicle drivers are assumed to take a 1-h lunch break between 11 a.m. and 1 p.m.

Two main capacity constraints are considered when creating a route: vehicle capacity and route capacity. Vehicle capacity is the maximum volume and weight that each vehicle can hold. Route capacity is the daily capacity for each driver: maximum number of stops, maximum number of lifts, maximum volume and weight that a driver can handle per day. WM business rules set the route capacity constraints and route time. Vehicle capacity dictates when a disposal trip should be made. If the vehicle capacity is larger than the route's volume capacity, there will be only one disposal trip and it will be the last visit before returning to the depot. If, however, the route's volume capacity is more than vehicle capacity, multiple disposal trips will be needed in a route. Since there are multiple disposal facilities available, careful selection of the best disposal is critical. Each vehicle is assumed to start from a depot and finish at the depot with zero volume. The typical problem size experienced at WM ranges between 10,000 and 54,000 commercial stops per depot per week. Thus, the daily problem size ranges between approximately 2000 and 10,000 per depot.

While minimizing the number of vehicles and total travel time is the major objective of VRPTW in the literature, the route compactness of a solution is very important to WM. The route compactness depends on how the stops are grouped into a route. A solution in which many routes cross over each other is less compact than one in which there are no overlapping routes. Balancing work among the vehicles is also important for implementation of a solution in the field. Our problem is summarized as follows.

Objectives:

- Minimize number of vehicles,
- Minimize travel time,
- Maximize route compactness,
- Balance workload among the vehicles.

Constraints:

- Time windows of stops and the depot,
- Vehicle capacity (i.e., volume, weight),
- Route capacity (i.e., the maximum number of lifts, volume and weight that can be handled per vehicle per day),
- Routing time limit per vehicle,
- Disposal trips (i.e., when a vehicle is full, it must go to a disposal facility),
- Driver's lunch break.

A mathematical programming model for a simplified version of the problem is presented in Sahoo et al. [1]. The VRPTW without considering multiple disposal trips and the lunch break itself is NP-hard, and finding a feasible solution with a fixed fleet size is an NP-complete problem (see Cordeau et al. [17]). Considering this and the large problem sizes, heuristic approaches are natural choice. We present our heuristic methods in the next two sections.

4. An extended insertion algorithm for waste collection VRPTW

In this section, we present a route construction algorithm that is an extension of Solomon's insertion algorithm [3]. The travel distance or travel time between any two stops is obtained by applying Dijkstra's shortest path algorithm to a GIS street network data. Since Solomon's insertion algorithm assumes that each vehicle departs from the depot, serves only one route, and returns to the depot, it should be extended to incorporate disposal operations and serving multiple routes and the lunch break.

The lunch break can be treated as a special stop that has a time window [11 a.m., 12 p.m.] with 1-h service time. The travel distance/time between the lunch break stop and a regular stop is assumed to be zero. However, the travel time from a stop before the lunch break to a stop after the break should be carefully considered. For example, if stops A and B are the stops before and after the lunch break, the travel distance/time from A to B should be added to the total travel time and be considered for the feasibility check of a route.

Below presents an extended insertion algorithm that considers multiple disposal trips and drivers' lunch breaks. All stops are marked as *unrouted* at the beginning of the algorithm (step 0). An empty route for a vehicle is created and is marked as the current route. The lunch-check of the route is marked as *not-yet*. A sub-route is initialized with a seed stop chosen based on either of Solomon's two starting criteria: farthest stop from the depot or the stop that has the earliest due time. While a route is initialized with a sequence [depot, the seed stop, depot] in Solomon's algorithm, a sub-route is initialized with a sequence [depot, the seed stop, lunch break, the closest disposal facility from the depot, depot] in the extended insertion algorithm (step 3). Since the route finally returns to the depot, the disposal facility that is closest from the depot is selected in an initial sub-route. Since the lunch break affects the feasibility of the afterward stops on the route, we explicitly insert it into the initial sub-route. Later on, if there is no need to have the break, i.e., the route finishes before the lunch time, it will be removed from the route.

Step 0: Mark all stops = *unrouted*
Step 1: If there is no *unrouted* stop,
  Go to step 6
Else,
  Go to step 2
Step 2: Start new route T, lunch-check of T = *not_yet*, reference = depot
Step 3: If a seed stop for a new sub-route CR can be selected,
    If lunch-check of T = *not_yet*,
      Initialize CR with the sequence (reference, the seed stop,
      lunch break, the closest disposal facility from the depot, depot)
    Else,
    Initialize CR with the sequence (reference, the seed stop,
      the closest disposal facility from the depot, depot)
  Else,
    Go to step 1
Step 4: Repeat insertion of *unrouted* stops to the current sub-route CR using Solomon's heuristic with time window constraints only until there is no feasible *unrouted* stops

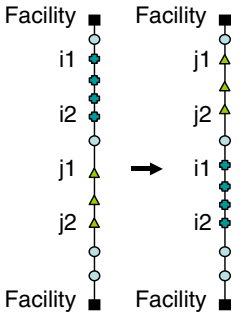

Fig. 5. The CORSS exchange for a single route.

Improve the sub-route CR by a simulated annealing (SA) meta-heuristic using the CROSS exchange local search method

Step 5: If the required capacity of CR is within the vehicle capacity,
Mark stops within CR as *routed*
Add CR to T
Go to step 1
Else,
Split CR into two sub-routes SR1 & SR2 based on the vehicle capacity constraint
Mark stops within SR1 as *routed,* and stops within SR2 as *unrouted*
Find the closest unloading stop from the last stop in SR1
Add the unloading stop and complete SR1, reference = the unloading stop
Add SR1 to T
Update the lunch-check of T
Go to step 3

Step 6: Improve the routes constructed using a SA meta-heuristic with CROSS exchange local search method

Step 7: Improve the routes by changing disposal facilities and their positions in the routes.

Step 8: Delete the lunch break from a route if the route can be completed before the break.

Step 9: Finish the algorithm.

After initialization, all stops that could be inserted without violation of time windows are repeatedly inserted using Solomon's insertion method (step 4). Then, the sub-route constructed is improved by a SA meta-heuristic using the CROSS exchange local search method proposed by Taillard et al. [6]. Although the CROSS exchange applies to two routes in their work, it can be applied to a single route as illustrated in Fig. 5. Two groups of consecutive stops are exchanged with each other. We randomly select the four indexes, $i1$, $i2$, $j1$, $j2$ with constraints $i1 \leqslant i2$, $i2 < i1 + 5$, $i2 < j1 \leqslant j2$, $j2 < j1 + 5$. In the constraints, 5 means the maximum number of consecutive stops for a group and the number was chosen arbitrarily with some experimental tests. Note that the vehicle capacity is not considered at this stage. Instead, only time window constraints of the vehicle and the stops are considered.

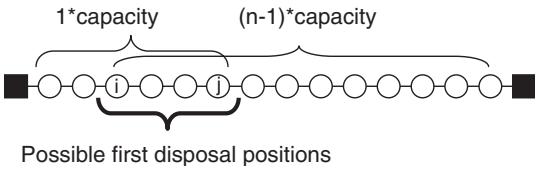

Fig. 6. Candidate positions for the first disposal trip.

After a feasible sub-route is constructed, the vehicle capacity is considered and the sub-route is split (step 5). If the current sub-route is within the vehicle capacity, it does not need to be split and is directly added to the current route. Otherwise, the cut position will be found based on the vehicle capacity. For example, if the current sub-route has the sequence [. . . , A, B, C, D, E . . .] and the vehicle can serve only up to stop C without exceeding the vehicle capacity, the cut position will be stop C. Then the stops whose positions are after the cut position in the sequence, are marked as *unrouted* stops. The current sub-route is modified and completed as [. . . , A, B, C, the closest disposal facility from stop C]. In order to reduce the total travel time, the disposal facility that is closest from the last stop C is selected. The disposal facility selected may or may not be the closest disposal facility from the depot. After completing the current sub-route, the completion time of the sub-route is calculated and the existence of the lunch break in the route is checked. The completion time of the sub-route, i.e., the time after disposal facility visit, will be the start time for the new sub-route of the current route. When the current sub-route includes the lunch break, the lunch-check of the current route is marked as *done*. The stops within the current sub-route are marked as *routed* and the completed current sub-route is added to the current route.

If there are no more *unrouted* stops remaining, the algorithm completes the route construction (step 1). Otherwise, it first attempts to create a new sub-route for the current route (step 3). If the algorithm cannot create a new sub-route due to the time constraint of the current vehicle, it starts a new route for another vehicle and creates a new sub-route for the route (step 2). When a new sub-route for the current route is created, a seed stop can be selected depending on the criterion. One way of selecting a seed stop is to find the closest *unrouted* stop from the disposal facility. Another way is to resume the previous sequence of stops in the previous sub-route. The former method is used in this research.

The new sub-route of the current route is initialized with a sequence [the disposal facility, the stop chosen, lunch break, the closest unloading location from the depot, depot] if the lunch-check of the route is *not-yet*, or with a sequence [the unloading location, the stop chosen, the closest unloading location from the depot, depot] if the lunch-check is *done*. Note that the order between the seed stop and the lunch break can be switched depending on the new sub-route's starting time. When the starting time is close to the lunch break's latest start time, the lunch break should be put before the seed stop. If a new route and its new sub-route are created, the sub-route is initialized with a sequence [depot, the stop chosen, lunch break, the closest disposal facility from the depot, depot] and its lunch-check is marked as *not-yet*. The processes above are repeated until all the stops are *routed*.

After a feasible solution is found, the routing times of the routes are again improved using a SA algorithm with CROSS exchange (step 6). Disposal facilities and their positions in the routes are revisited and changed if those changes shorten the travel time (step 7). Candidate positions in a route for disposal facilities can be found recursively. Fig. 6 shows the way finding the candidate positions, i through j, for the first disposal facility. When the route requires n disposal trips based on the route's load and the vehicle's capacity, position i is the earliest position that the total load (volume) from which to the end

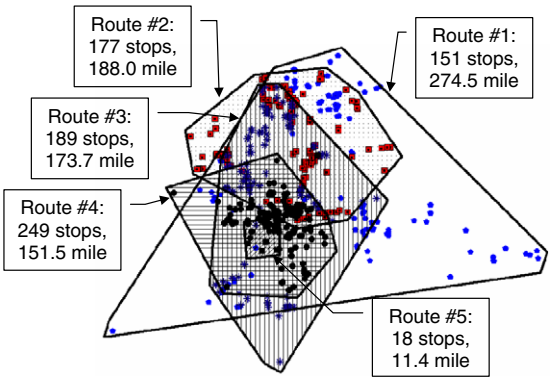

Fig. 7. Solution instance of the extended insertion waste collection VRPTW algorithm.

of the route is within $(n-1)^{*}vehicle\ capacity$. The latest possible position j for the first disposable trip is the latest one that the load amount from the first stop to which is within the vehicle capacity. After the first disposal position is determined, the candidate positions for the second disposal trips can be determined with the same manner and so on. These procedures enumerate all the possible combinations of the disposal trip positions. Then, the route is evaluated with the best disposal facility for each position for every possible combination of disposal trip positions and the best route will be chosen. Although we use a full enumeration method, the combinations are not unmanageably large because typical number of disposal trips is 2 or 3 and the last disposal trip position is fixed right before the depot. Since this is still a computationally expensive operation, however, we provide an option that user can decide whether to use it or not. More efficient algorithm such as the splitting algorithm of Prins [19] can be used for this operation. This is one of our future research areas.

Finally, the necessity of the lunch break of each route is examined and the lunch break is deleted if it is not necessary for the route (step 8). When the route can be completed before the lunch break's latest start time, the break is deleted from the route.

4.1. Problems of the insertion algorithm

Although we can solve real world problems using the insertion algorithm, it has prominent weak points from a practical perspective. The first problem is the route compactness of the solutions. Fig. 7 shows the solution generated by the insertion algorithm for a test problem. The problem requires five vehicles and each polygon represents a vehicle route. For clear presentation, depot and disposal facility locations are not shown. As shown in Fig. 7, the solution has many overlaps over the route areas. This kind of solution may generate difficulty to manage the operation in practice. In fact, we proposed solutions for several problems using the insertion algorithm approach to field managers of WM but they did not accept them because of the overlapping issue.

The second problem involves workload balancing. Simply by following the insertion algorithm, we have experienced many cases in which a solution has routes covering just a few stops. The routes constructed late by the algorithm usually have smaller number of stops than the routes constructed earlier. For example, when a stop is left after constructing a route, the stop itself would form another route. In Fig. 7, route 5

has only 18 stops while the others have more than 150 stops. Again, the management and the drivers of WM do not accept this unbalanced routing schedule.

The third problem addresses the computation time. Although the insertion algorithm is much faster than other algorithms like the parallel insertion algorithm and tabu search, its computational time is still longer than desired in practice, especially for large problems. Step 4 of the algorithm requires checking the combination of every possible position of the route and every possible stop. While Solomon's benchmark problems contain 100 stops, the problems considered in this research may have more than 2000 stops. When the current route being constructed has 100 stops and there are 1000 unrouted stops, the next insertion of a stop requires checking 100,000 different combinations and each combination requires feasibility check and travel time calculation.

In order to handle these problems, a clustering-based waste collection VRPTW algorithm is developed and presented in the next section.

5. A clustering-based waste collection VRPTW algorithm

The steps of the algorithm are shown below. First, the number of vehicles required is estimated by the total workload divided by the daily workload capacity for each vehicle (step 0). Here, the total workload is determined by the total number of stops and total volume. Note that we assume that the vehicles are of the same type.

Step 0: Estimate the number of vehicles, N, based on the total workload
Step 1: Generate capacitated N clusters
Improve the cluster (route) compactness
Index the clusters in decreasing order based on the number of stops of the cluster
Set all the clusters as *not_finalized*.
Step 2: Repeat until all the clusters are *finalized* or all the stops are *routed*
For each cluster $n = 1 \ldots N$,
Construct a route for the cluster using the extended insertion algorithm
If any stops in the cluster is remained as *unrouted*,
Reassign the stops to the closest cluster among the *not_finalized* clusters
Set cluster *n finalized*.

Step 3: If there are *unrouted* stops remained,
$N = N + 1$
Go to step 1
Else,
Go to step 4
Step 4: Improve the route compactness by an inter-route improvement algorithm
Step 5: Improve the route time of each route by an inter-route improvement algorithm
Step 6: Finish the algorithm.

Then, the stops are clustered using a capacitated clustering algorithm, which is a variant of the K-means algorithm (step 1). The clustering algorithm works as follows. As in the K-means algorithm, N initial centroid seed stops are selected randomly as the first step and the remaining stops are assigned to the

nearest centroid seed stop. Then, a new centroid of each cluster is calculated. The next step is different from the standard K-means algorithm. In the standard K-means algorithm, the stops are clustered by considering only the distances between the stops and the centroids, i.e., a stop is assigned to the cluster whose centroid is the closest to the stop. In our capacitated clustering algorithm, we also consider the centroid of the centroids, which would be called 'the grand centroid'. The stops are sorted in descending order based on the distances between the grand centroid and the stops. The farthest stop from the grand centroid is assigned first to the nearest centroid (cluster). Then, the next farthest stop and so on. When a stop is assigned to a centroid or a cluster, the capacity of the cluster is considered. The capacity of the cluster is defined by the number of stops contained in the cluster, the total volume of the stops in the cluster, and the estimated route time between the stops in the cluster. The route time is the sum of travel time and service time. The travel time of each cluster is estimated by a simple traveling salesman problem algorithm. When a cluster whose centroid is the nearest from a stop already has reached its capacity, the stop would be assigned to another cluster. Through this process, capacitated clusters can be obtained. The steps above: obtaining centroids and the grand centroid, sorting the stops, and assigning stops to the clusters, are repeated until there is no change.

Another factor that can be included in assigning a stop to a cluster is the time window for the stop. When considering to add a stop to a cluster, if the stop time window conflicts with other stops already assigned to the cluster, then the stop should be assigned to a different cluster. For example, if stop A has time window [10:30 a.m., 11:00 a.m.] and 20-min service time, stop B has time window [11:00 a.m., 11:30 a.m.] and 20-min service time, and the travel time between the two is 50 min, the two stops cannot be assigned to the same cluster because the two cannot be served within their time windows using the same vehicle. The objective of the clustering is to have capacitated clusters in which a vehicle can cover all the stops in a cluster.

When there are overlaps among the clusters constructed, the clusters are adjusted by a simple move improvement algorithm. The basic idea of the improvement is very simple: if a stop is assigned to a cluster that is not the nearest one from the stop in terms of the distance between the stop and clusters' centroid, move the stop to the nearest cluster when its move does not violate the clusters' capacity constraints (volume, number of stops, and estimated routing time) and when its move improves the overall route quality. In order to quantify the route (or cluster) compactness, a shape metric S_m is devised as Eq. (1). ST_c is the center location of a route when the distance measure is Euclidean or Manhattan, and is the closest stop to the centroid of each Route R_j when the distance measure is the street distance. In our computational experiment, we use the Euclidean distance. A cluster result with smaller S_m is better than one with larger S_m with regards to route compactness.

$$S_m = \sum_{\substack{\text{All tours } T_i \\ \text{in a solution}}} \sum_{\substack{\text{All routes } R_j \\ \text{in tour } T_i}} \sum_{\substack{\text{All stops } ST_k \\ \text{in route } R_j}} \text{Dist}(ST_k, ST_c). \tag{1}$$

In addition to the shape metric, another measure N_h is devised and is used to measure the cluster overlap. Each cluster forms a convex hull as presented in Fig. 8 using Graham's [18] convex hull algorithm. The measure N_h counts the number of stops that belong to more than one convex hull of clusters. While N_h of Fig. 8a is 2, N_h of Fig. 8b is 0. Obviously, Fig. 8b clustering is better than Fig. 8a with regards to cluster overlapping. We also consider balancing of route times among the clusters when we improve stop-cluster assignment. Although the algorithm attempts to remove all the overlaps, it cannot guarantee no-overlap because of multiple constraints.

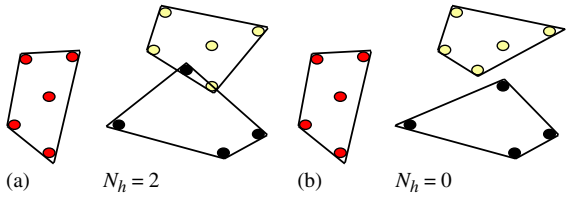

Fig. 8. Convex hulls and measure N_h.

After the capacitated N clusters are obtained, the clusters are sorted in descending order of the number of stops contained in it. All the clusters are set as *not_finalized*. Then, a route for each cluster, from cluster index number 1 to cluster index number N, is constructed using the extended insertion algorithm (step 2). After a route is constructed for a cluster, *unrouted* stops in the cluster, if any, are reassigned based on centroid to the closest cluster among the *not_finalized* clusters. Although we intend to have a route cover all the stops of a cluster through the clustering algorithm, it is not always possible because of the complexity of the problem including time windows and travel time. Thus, we need this reassignment step. These procedures are repeated until all the clusters are *finalized* or all the stops are *routed*. A cluster becomes *finalized* when it reaches its capacity and is not able to take any more stops.

If there are *unrouted* stops after all the routes reach their capacities, this means that a feasible solution with the calculated number of vehicles has not been found. Then, the number of vehicles is increased and the process is repeated beginning with clustering (step 3). When a feasible solution is found, the solution can be further enhanced by improving the route compactness using an inter-route improvement algorithm (step 4) and by improving the travel time of each route (step 5). Both improvement algorithms use SA with CROSS exchange local search method. The CROSS exchange operator is applied to two routes in step 4 and is to one route in step 5. When a new solution generated by the operator is feasible, it will be accepted by the program with our SA procedure. Both steps are iterated given number of times or until there is no more improvement or until given computation time.

6. Computational results

The algorithms are implemented in C language. Computational times in this section are based on a Microsoft Server 2003 machine of 3.05 GHz Xeon processor with 1.0 GB of RAM. The machine is configured such that its computational power can be allocated up to 50 percent of its full power to a single application. Fig. 9 shows the solution generated by the clustering-based waste collection VRPTW algorithm for the problem shown in Fig. 7. As shown in the figure, the solution has better route compactness, i.e., less overlapping, than the solution of the extended insertion algorithm shown in Fig. 7. Additionally, the algorithm shows improvement with regards to the workload balancing. Table 1 compares the algorithm with the insertion algorithm. In order to show the impact of SA improvement algorithm, the results of without SA are also presented. The columns show the following statistics:

- V_n: number of vehicles required.
- S_m (mile): the shape metric defined in the previous section.
- N_h: the number of stops that belong to more than one convex hull of clusters.

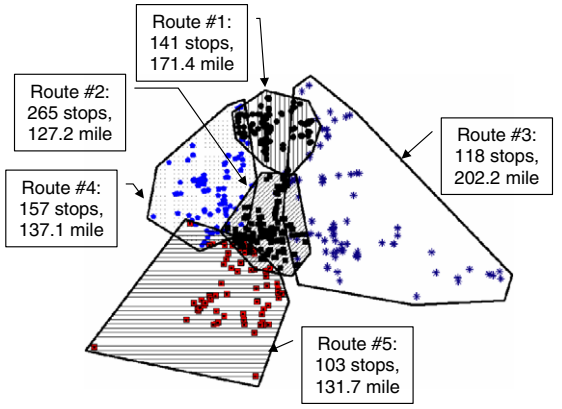

Fig. 9. Solution instance of the clustering-based waste collection VRPTW algorithm.

Table 1
Solution results for the example problem (804 stops)

Algorithm	V_n	S_m (mile)	N_h	*TD* (mile)	*RTD* (sec)	*CT* (sec)
Extended insertion without SA	5	3522.2	646	807.3	32742	125
Extended insertion with SA	5	3793.5	675	798.9	32741	154
Cluster based without SA	5	2490.4	314	856.7	8430	43
Cluster based with SA	5	2350.4	264	769.5	18047	92

- *TD* (mile): total travel distance of the solution.
- *RTD* (second): the route time deviation defined by the difference between the longest route and the shortest route with regard to time.
- *CT* (second): computational time.

The clustering-based algorithm outperforms the insertion algorithm in the aspects of the route compactness (S_m), route overlap (N_h), and workload balancing (*RTD*) while it requires significantly less computing time (*CT*). The route compactness of the solutions can be easily compared by the figures (Fig. 7 vs. Fig. 9) and also as shown in Table 1, the shape metric S_m and route overlap N_h of the solution generated by the clustering-based algorithm is much smaller than the solution's generated by the insertion algorithm. The difference of the total travel distance (*TD*) is not significant in this example. However, the improvement of travel distance by using SA is significant, particularly in the clustering-based algorithm (10.2 percent improvement). The SA also affects other measures. Note that our problem has multiple objectives and they are conflictive in some cases. We generally set the importance of objectives as (1) S_m and N_h, (2) V_n, (3) *TD*, and (4) *RTD*.

Since there are no publicly available benchmark problems of the waste collection VRPTW, we extracted several problems from the real world problems and put them on http://www.postech.ac.kr/lab/ie/logistics/WCVRPTW_Problem/benchmark.html for other researchers in this area. Each problem has four lines of basic information, one line of stop data header, and multiple lines of stop data. Four lines of basic

Table 2
Waste VRPTW benchmark problem sets

Problem set	Number of stops	Vehicle cubic yards capacity	Route cubic yards capacity/day	Route max stop count/day	Depot operating hours
102_stop.txt	102	280.0	400.0	500	11
277_stop.txt	277	200.0	2200.0	500	9
335_stop.txt	335	243.0	400.0	500	9
444_stop.txt	444	200.0	400.0	500	9
804_stop.txt	804	280.0	10000.0	500	11
1051_stop.txt	1051	200.0	800.0	500	13
1351_stop.txt	1351	255.0	800.0	500	13
1599_stop.txt	1599	280.0	800.0	500	11
1932_stop.txt	1932	462.0	2000.0	500	14
2100_stop.txt	2100	462.0	2000.0	500	11

information include vehicle capacity, total yards allowed for a vehicle per day, total number of stops allowed for a vehicle per day, lunch break time in seconds, default speed in mile per hour, and stop data. A vehicle cannot handle more than the total yards allowed for a vehicle or the total number of stops allowed. The default speed in mile per hour is for OD matrix construction. Although we apply a shortest-path algorithm to GIS to obtain OD matrix, i.e., travel time between any two stops, it is hard to publish those ODs on the Internet because of their sizes. For example, the size of the OD matrix for 1599_stop.txt is about 80 MB. Thus, we recommend readers to use the default speed provided and the Manhattan distance for the calculation of OD matrix. For example, the travel time between stop 0 (x-coordinate = 1215029, y-coordinate = 3461398) and stop 1 (1229125, 3460107) in 102_stop.txt is 262.3 s. The Manhattan distance between them is 15,387 f and it is 2.914 mile. Since the default speed is 40 MPH, the travel time is 2.914 mile $\times$ 3600 s/40 mile, which is 262.3 s. Our computational results in this section is also based on this method of OD calculation.

Individual stop has information of stop id, x-coordinate in feet, y-coordinate in feet, earliest service starting time in HHMM format, latest service starting time in HHMM format, service time in seconds, load in yards, and stop type. In HHMM format, the first two digits show hour and the remaining digits show minute. For example, 1030 stands for 10:30 a.m. and 1445 stands for 2:45 p.m. The depot has stop type of 0, a landfill or transfer station has stop type 2, and a regular stop has stop type 1. Table 2 shows the characteristics of the problems sets.

Table 3 shows the results of our algorithms. Note that SA is used in the algorithms. The first column of heading A shows the algorithm used: 1 for extended insertion with SA, 2 for clustering based with SA. The other column headings are identical to those of Table 1. As shown in Table 3, while the number of vehicles obtained by the clustering-based algorithm is sometimes larger than the number required by the insertion algorithm, its solution is almost always better in other aspects. More importantly, the route compactness of its solution is always better.

The clustering-based algorithm with SA is embedded in WasteRoute^{TM}, a comprehensive enterprise-wide web-based route management application that took into account the specific routing considerations of WM. WasteRoute^{TM} was deployed across North America beginning March 2003. While the exact savings that were experienced varies, one specific example in Elgin, IL shows the reduction of one

Table 3
Computational results

Problem set	A	V_n	S_m (mile)	N_h	TD (mile)	RTD (sec)	CT (sec)
102_stop.txt	1	2	869.4	55	218.4	22296	2
	2	3	399.1	0	205.1	9903	3
277_stop.txt	1	3	858.0	208	521.9	13843	37
	2	3	385.9	104	527.3	4201	10
335_stop.txt	1	5	410.4	225	191.1	2377	28
	2	6	243.3	0	205.0	8983	11
444_stop.txt	1	10	70.0	291	82.9	2422	372
	2	11	28.5	1	87.0	2636	16
804_stop.txt	1	5	3793.5	675	798.9	32741	154
	2	5	2350.4	264	769.5	18047	92
1051_stop.txt	1	17	6790.5	981	2874.0	43556	682
	2	18	3615.5	107	2370.4	20471	329
1351_stop.txt	1	7	5184.0	1095	1139.0	17391	3297
	2	7	2145.9	64	1039.7	9127	95
1599_stop.txt	1	13	9979.1	1452	24276.0	1775.0	298
	2	13	4141.3	1300	11666.0	1459.2	212
1932_stop.txt	1	15	8060.1	1819	1524.9	31649	318
	2	17	3166.5	1079	1395.3	25833	424
2100_stop.txt	1	16	14562.7	1874	2401.2	20913	602
	2	16	5761.8	1155	1833.8	5764	408

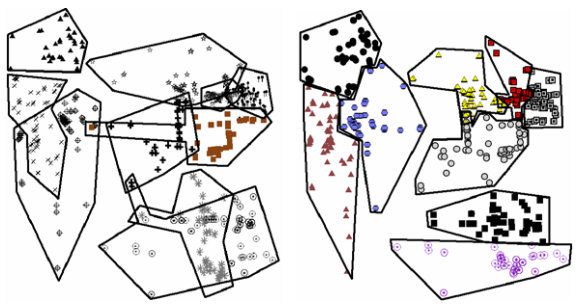

Fig. 10. An implementation example.

route through WasteRoute^{TM} (Fig. 10). Before WasteRoute^{TM}, that area required ten 9-h routes and its productivity was 57.06 yards/h. With the solution of WasteRoute^{TM}, that area now requires nine 9-h routes and its productivity is 63.40 yards/h. Each polygon shows individual route. Using WasteRoute^{TM}, WM could reduce 984 routes at the end of the year, resulting in a savings of $18 million. Estimated over the

course of the entire year for 2004, this translates to a savings of $44 million. Since WasteRoute^{TM} is still being deployed to various sites throughout the continent, additional route reductions will add to the savings already being realized. The processes and outcome of the entire project of WasteRoute^{TM} are discussed in [1].

7. Conclusions

In this paper, we have discussed a waste-collection VRPTW with consideration of multiple disposal trips and drivers' lunch break. We presented its simplified mathematical model. An extended insertion algorithm and a clustering-based waste collection VRPTW algorithm are developed and implemented in a software system for the problem. Using the clustering-based waste collection VRPTW algorithm, we could handle the important practical issues such as route compactness and territory, workload balancing, and computational time. Solutions based on the proposed approach have been successfully deployed at the client sites. The processes and outcome of the entire project are discussed in Sahoo et al. [1]. A set of waste collection VRPTW benchmark problems is also presented. Future research directions include on-call dynamic demand handling and application of the proposed algorithms to non-waste collection industries.

References

[1] Sahoo S, Kim S, Kim B-I, Kraas B, Popov Jr. A. Routing optimization for waste management. Interfaces 2005;35(1):24–36.
[2] Golden BL, Assad AA, Wasil EA. Routing vehicles in the real world: applications in the solid waste, beverage, food, dairy, and newspaper industries. In: Toth P, Vigo D, editors. The vehicle routing problem. Philadelphia, PA: SIAM; 2002. p. 245–86.
[3] Solomon MM. Algorithms for the vehicle routing and scheduling problem with time window constraints. Operations Research 1987;35(2):254–65.
[4] Potvin JY, Rousseau JM. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. European Journal of Operations Research 1993;66:331–40.
[5] Rochat Y, Taillard ED. Probabilistic diversification and intensification in local search for vehicle routing. Journal of Heuristics 1995;1:147–67.
[6] Taillard ED, Badeau P, Gendreau M, Guertin F, Potvin JY. A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation Science 1997;31(1):170–86.
[7] Taillard ED, Laporte G, Gendreau M. Vehicle routing with multiple use of vehicles. Journal of the Operational Research Society 1996;47:1065–70.
[8] Weigel D, Cao B. Applying GIS and OR techniques to solve Sears technician-dispatching and home-delivery problems. Interfaces 1999;29(1):112–30.
[9] Tung DV, Pinnoi A. Vehicle routing-scheduling for waste collection in Hanoi. European Journal of Operational Research 2000;125:449–68.
[10] Poot A, Kant G, Wagelmans APM. A savings based method for real-life vehicle routing problems. Journal of the Operational Research Society 2002;53:57–68.
[11] Angelelli E, Speranza MG. The periodic vehicle routing problem with intermediate facilities. European Journal of Operational Research 2002;137:233–47.
[12] Angelelli E, Speranza MG. The application of a vehicle routing model to a waste-collection problem: two case studies. Journal of the Operational Research Society 2002;53:944–52.
[13] Teixeira J, Antunes AP, Sousa JP. Recyclable waste collection planning—a case study. European Journal of Operational Research 2004;158:543–54.

[14] Eisenstein DD, Iyer AV. Garbage collection in Chicago: a dynamic scheduling model. Management Science 1997;43(7): 922–33.
[15] Chang N-B, Lu HY, Wei YL. GIS technology for vehicle routing and scheduling in solid waste collection systems. Journal of Environmental Engineering 1997;123:901–33.
[16] Mourao MC, Almeida MT. Lower-bounding and heuristic methods for a refuse collection vehicle routing problem. European Journal of Operational Research 2000;121:420–34.
[17] Cordeau J-F, Desaulniers G, Desrosiers J, Solomon MM, Soumis F. VRP with time windows. In: Toth P, Vigo D, editors. The vehicle routing problem. Philadelphia, PA: SIAM; 2002. p. 157–93.
[18] Graham RL. An efficient algorithm for determining the convex hull of a finite planar set. Information Processing Letters 1972;1:132–3.
[19] Prins C. A simple and effective evolutionary algorithm for the vehicle routing problem. Computers and Operations Research 2004;31:1985–2002.