Vehicle Routing and Scheduling Problems for Convenience Store Industry Considering Soft Time Windows

Narath BHUSIRI [a], Ali Gul QURESHI [b], Eiichi TANIGUCHI [c]

[a,b,c] *Department of Urban Management, Kyoto University, Katsura, Nishikyo-ku, Kyoto 615-8540, Japan*
[a] *E-mail: bhusiri.narath.65w@st.kyoto-u.ac.jp*
[b] *E-mail: aligul@kiban.kuciv.kyoto-u.ac.jp*
[c] *E-mail: taniguchi@kiban.kuciv.kyoto-u.ac.jp*

Abstract: Vehicle routing problem with soft time windows and simultaneous pickups and deliveries (VRPSTWSPD), which illustrates a class of distribution and routing systems of convenience store industry, is introduced. The VRPSTWSPD is a novel variant, where it combines characteristics of two existing variants in the literature. A mathematical model of the VRPSTWSPD is first developed in the paper. A branch-and-price exact-based procedure with a two-stage subproblem solution algorithm for solving small-sized instances is then described, followed by an efficient genetic algorithm metaheuristic-based procedure taking into account intermediate infeasible search strategy, which is more adequate for larger instances. Both solution algorithms were successfully tested on benchmark instances. The exact solutions can be used as references to evaluate the performances of the proposed genetic algorithm.

Keywords: Vehicle Routing Problem, Soft time windows, Simultaneous Pickups and Deliveries, Column Generation, Genetic Algorithm

1. INTRODUCTION

Convenience store chain such as Lawson, FamilyMart, and 7-Eleven has played a significant role in retail business. As reported by the U.S. Agricultural Trade Office (2011), for instance, convenience store industry in Japan represented approximately 25% (in 2009) and 26% (in 2010) of Japanese's food retail sales. Due to its extensive investment, a wide variety of services, and ease of accessibility, convenience store industry has continuously grown and expanded. This consequently brings a great challenge to logistician's attention in order to design and develop efficient schemes for the corresponding distribution and routing systems.

One class of the distribution and routing systems of convenience store industry is based on a centralized system, where all products from suppliers are delivered to convenience stores via distribution center. No direct delivery from supplier is allowed; rather deliveries are made by internal distributors. All convenience stores are treated in a franchise network. A distribution center, denoted as depot, is a place where a fleet of service vehicles is housed, all merchandises from suppliers are stored, and a set of reused/unwanted items is preliminary kept for further processing. Each vehicle starts (and ends) the day at depot and further visits a subset of franchising convenience stores to give services. The services are bi-directional activities in which delivery of goods and collection of reused/out-of-date items are made simultaneously at each visit. In order to satisfy Just-In-Time concept, all franchising convenience stores attempt to minimize their waste, e.g. excess inventory stock and unnecessary manpower. During the shift work, only a few staffs are then required to work at

each franchising store (Ngaochay and Walsh, 2011) so that most stores might prefer to be serviced within their specific time-intervals (called time windows) to avoid heavy loads at peak hours.

In distributor point of view, however, strictly following franchising convenience store's time windows is a difficult task. To prevent delayed services, a large number of service vehicles is a must, and consequently overall distribution costs are high. On the other hand, additional concerns arise if delivery vehicles arrive too early. Since parking spaces are often not provided in many convenience stores, especially ones located in dense areas, waiting places must therefore be determined. Parking at inappropriate place might lead to traffic congestion, while parking at appropriate place might be more costly, i.e. due to parking fee. Besides, such waiting is a non-beneficial activity and firm's resources are underutilized. To overcome these kinds of problems, the time window at each franchising convenience store could rather be violated and appropriate penalty must be paid when untimely visit occurs.

The problem addressed above can be characterized as the vehicle routing problem with soft time windows and simultaneous pickups and deliveries (VRPSTWSPD). The VRPSTWSPD is an extension to the well-known vehicle routing problem (VRP) (Baldacci *et al*., 2012; Laporte, 2009; Taniguchi *et al*., 2001; Toth and Vigo, 2002). It aims to design a set of feasible routes to fulfill all requests such that all of the following are respected: 1) Minimization of overall distribution costs including vehicle utilized costs, travel costs, and penalties, 2) Minimization of waiting time of all vehicles, and 3) Attaining the acceptable level of satisfaction. The feasibility conditions of each route rely heavily on capacity constraints, assignment constraints, working hour constraints, and soft time window constraints.

In time window setting, illustrated in Figure 1, the hard time window of each franchising convenience store, which is indexed by i, is represented by $[a_i,b_i]$; if a vehicle arrives within $[a_i,b_i]$, the service must start immediately without penalty. In particular, the soft time window $[a'_i,b'_i]$, obtained by considering the trade-off between fixed vehicle cost and time-dependent arrival penalties, is also defined (definitions of a'_i and b'_i will be given later). The vehicle is possible to arrive and start servicing at franchising convenience store after time window (but not later than b'_i) with taking late arrival penalty into account. On the other hand, if the vehicle arrives early, it must wait to begin the service until a_i with taking waiting cost (early arrival penalty) into account. Arrival beyond $[a'_i,b'_i]$ is clearly infeasible. The vehicle is not permitted to wait if it arrives within time window or later, and the vehicle must leave as soon as it completes the service. An appearance of maximum violated hard time window, i.e. a'_i and b'_i, is inevitable as allowing too long waiting or too long delay is always impractical.

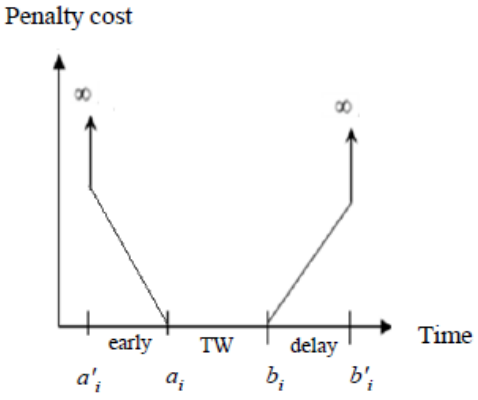

Figure 1. Soft time window setting at franchising convenience store

The VRPSTWSPD is a novel variant in the VRP literature. It is a combination of the vehicle routing problem with simultaneous pickups and deliveries (VRPSPD), and the vehicle routing problem with soft time windows (VRPSTW). In this paper, we first introduce 3-index mixed integer programming model of the VRPSTWSPD. To the best of our knowledge, no VRPSTWSPD model has been published so far. We then propose a branch-and-price exact-based approach to optimally solve small-sized instances of the VRPSTWSPD. Decomposition of the VRPSTWSPD results in a new pricing subproblem, where a two-stage subproblem solution technique is consequently developed to solve it. We also propose an efficient genetic algorithm (GA), which is more adequate and favorable for medium-sized and large-sized instances. The proposed GA considers an advanced search strategy, i.e. an intermediate infeasible search, for the purpose of more advanced exploration of the solution space. Last but not the least, exact solutions obtained from the branch-and-price approach can be set as references to evaluate the performances of the proposed GA.

The rest of the paper is organized as follows: Section 2 reviews some related works. Section 3 presents mathematical formulation of the VRPSTWSPD. Sections 4 and 5, respectively, propose the exact approach and the metaheuristic approach to solve the VRPSTWSPD. Section 6 reports numerical results, while section 7 concludes the paper.

2. LTERATURE REVIEW

As mentioned, the VRPSTWSPD is generalized from the VRPSTW and the VRPSPD. Both the VRPSPD and the VRPSTW are NP-hard problems, so is the VRPSTWSPD. The VRPSPD has emerged in the literature due to Min (1989) in which a practical problem of public library distributing system was investigated. The author also developed a heuristic technique comprising three decision stages and evaluated it with a manual solution method. Later, Salhi and Nagy (1999) stated the definitions of the VRPSPD and related problems. All problems were then clearly categorized, and insertion heuristic methods were developed to solve the subject problems.

Dethloff (2001) particularly addressed importance of the VRPSPD in the new logistics context, where an integration of forward and reverse flows (reverse logistics) has been made due to eco-friendly concept. A mathematical model, some innovative insertion heuristics as well as a set of benchmark instances were extensively developed in his work. Since the work of Dethloff, a great attention has been paid to the VRPSPD, hence resulting in a wide exploration of efficient solution approaches. Such approaches can be sorted as 1) Heuristics (e.g. Nagy and Salhi, 2005; Subramanian and Cabral, 2008), 2) Metaheuristics including genetic algorithm (e.g. Tasan and Gen, 2012; Wang and Chen, 2012), tabu search algorithm (e.g. Bianchessi and Righini, 2007; Montane and Galvao, 2006), ant colony system (e.g. Gajpal and Abad, 2009), and particle swarm optimization (e.g. Ai and Kachitvichyanukul, 2009), and 3) Exact algorithms (e.g. Dell'Amico *et al*., 2006; Subramanian *et al*., 2010b). The comprehensive surveys of the VRPSPD and its extensions can be found in Parragh *et al*. (2008) and Subramanian *et al*. (2010a).

As opposed to the VRPSPD, a body of relevant literature on the VRPSTW is relatively scant. Early works on the VRPSTW with use of simple heuristics have been approximately launched in 1990s. Such works might include the work of Balakrishnan (1993) in which three simple heuristics have been presented to solve the VRPSTW. Later, Taillard *et al*. (1997) and Gendreau *et al*. (1999) have proposed the new problem definition of the VRPSTW, where penalty must go into effect when late arrival arises, while early arrival is possible at no

penalty. Tabu search heuristics were employed in their works. Ioachim *et al.* (1998) also focused on the shortest path problem taking into account time windows and time costs, where the problem is equivalent to the pricing subproblem of the VRPSTW. Since the beginning of the 21st century, the studies on heuristic and metaheuristic procedures to tackle the VRPSTW have been more and more carried out (see e.g. Ando and Taniguchi, 2006 (genetic algorithm); Chiang and Russell, 2004 (tabu search); Fu *et al.*, 2008 (tabu search); Figliozzi, 2010 (iterative construction algorithm and route improvement algorithm); Hashimoto *et al.*, 2006 (local search algorithm); Ibaraki *et al.*, 2005 (local search algorithm); Ioannou *et al.*, 2003 (nearest-neighbor heuristic); Taniguchi and Kakimoto, 2003 (genetic algorithm); Taniguchi and Shimamoto, 2004 (genetic algorithm)).

To date, use of exact algorithms has been in turn successful in solving the VRPSTW to optimality (see e.g. Liberatore *et al.*, 2011; Qureshi *et al.*, 2009, 2010, 2012; Tagmouti *et al.*, 2007; Westphal and Krumke, 2008). Size of instances being successfully solved by exact approaches varies from small-sized to large-sized, depending on the effectiveness of algorithms and the difficulty of problem characteristics.

In this paper, an exact-based branch-and-price algorithm is employed to solve the VRPSTWSPD to optimality. Decomposition of the VRPSTWSPD leads to a newly defined pricing subproblem, where various complex constraints are composed of. Hence, a two-stage solution algorithm is also developed for the novel subproblem. As a result of NP-hard problem of the VRPSTWSPD, an efficient GA with newly advanced search strategy is also employed. Several significant genetic operators are also presented to enhance the performance of the proposed GA.

3. THE VRPSTWSPD MODEL

The VRPSTWSPD model is generalized from the VRPSPD model (Ai and Kachitvichyanukul, 2009) and the VRPSTW model (Qureshi et *al.*, 2009). Let us first denote K as a fleet of homogeneous service vehicles, and C as a set of franchising convenience stores with known demands. Note that term "franchising convenience store" will hereafter be referred to as "customer". The VRPSTWSPD can be defined on a directed graph $G=(V,A)$, where vertex set V contains depot (numbered 0) and set C, and arc set A comprises all feasible arcs (i,j), $i,j \in V$, and $i \neq j$. Each arc $(i,j) \in A$ is associated with a travel cost c_{ij} and a travel time t_{ij}. The duration of service st_i is also included in the travel time t_{ij}. For the sake of simplicity of the problem, both c_{ij} and t_{ij} are integer and follow triangular inequalities. Each vertex $i \in C$ has a non-negative pickup request u_i and a non-negative delivery request d_i. At depot, service time, pickup demand, and delivery demand are fixed at zero. Depot time window $[a_0,b_0]$ is also given, signifying the operating hours of the depot. A vehicle $k \in K$, which starts and ends at depot, can cover only one route; therefore terms "vehicle" and "route" will hereafter be used interchangeably. Each vehicle is assigned a fixed capacity Q, and a fixed vehicle cost fc, representing its daily operating cost, maintenance cost, and labor cost. Vehicle capacity must always be respected along its entire route. The fixed vehicle costs are easily incorporated by adding them to the travel costs of all outgoing arcs from depot, i.e. in c_{0j}, $j \in C$. The VRPSTWSPD model presented in this paper is deterministic version so that all values are constant and known in advance of the planning.

At vertex $i \in C$, a hard time window $[a_i,b_i]$ is extended by a_i to a'_i ($a'_i \leq a_i$) and b_i to b'_i ($b_i \leq b'_i$). Thus, the time window involved becomes $[a'_i,b'_i]$, where a'_i specifies the maximum allowable waiting time of the vehicle to arrive and wait to service until a_i with early arrival penalty, while b'_i is the extended latest possible service start time with late arrival penalty. The

soft time window $[a'_i, b'_i]$, $\forall i \in C$ must always respect depot time window. Let c_e and c_l be unit early arrival penalty and unit late arrival penalty, respectively (generally $c_e < c_l$). To obtain a'_i, a trade-off between fixed vehicle cost and early arrival penalty is considered. That is to say, penalty is acceptable when it is not more costly than cost of an additional vehicle. Otherwise, having one more vehicle to serve a particular customer is a better choice. Calculation of a'_i can be expressed as follows:

$$a'_i = a_i - \min[(a_i - a_0), (\frac{c_{0i} + c_{i0}}{c_e})] \qquad \forall i \in C \tag{1}$$

The upper limit of delayed service b'_i comes from two sources. First is a negotiated value with each customer. While the other one is to consider a trade-off between fixed vehicle cost and late arrival penalty, which is similar to that of a'_i. The minimum one of these two is then selected. Let ne_i be the maximum limit of late arrival time agreed by a customer $i \in C$. The b'_i is determined as follows:

$$b'_i = \min\{(ne_i), (\min[(b_0 - t_{i0}), (b_i + (\frac{c_{0i} + c_{i0}}{c_l}))])\} \qquad \forall i \in C \tag{2}$$

Note that an appearance of ne_i, $i \in C$ in equations (2) is essential in order to maintain the quality of service as it ensures that the service is never delayed beyond acceptable time limit specified by customer. The VRPSTWSPD is formulated as 3-index vehicle flow model. It requires four groups of decision variable. First group is non-negative time variables s_{jk} indicating service start time at vertex $j \in C$ by vehicle $k \in K$. The second group is binary arc flow variables x_{ijk} having value one if vehicle $k \in K$ traverses the arc $(i,j) \in A$ and zero otherwise. The third group is non-negative integer variables y_{ijk} representing load of vehicle $k \in K$ when traversing the arc $(i,j) \in A$. Last group is non-negative time variables l_k indicating departure time of vehicle $k \in K$ from the central depot.

In problem definition, penalty is particularly based on arrival time of vehicle rather than service start time. Moreover, service start time is also conditional to arrival time of vehicle. Let p_{jk} denotes arrival time of vehicle $k \in K$ at vertex $j \in V$, the modified time-dependent travel cost c'_{ij} on arc $(i,j) \in A$ and the service start time variables s_{jk} at $j \in C$ by $k \in K$ can be defined as follows:

$$c'_{ij} = \begin{cases} \infty & if \quad p_{jk} < a'_j \\ c_{ij} + c_e(a_j - p_{jk}) & if \quad a'_j \le p_{jk} < a_j \\ c_{ij} & if \quad a_j \le p_{jk} \le b_j \\ c_{ij} + c_l(p_{jk} - b_j) & if \quad b_j < p_{jk} \le b'_j \\ \infty & if \quad p_{jk} > b'_j \end{cases} \tag{3}$$

$$s_{jk} = \begin{cases} \infty & if \quad p_{jk} < a'_j \\ a_j & if \quad a'_j \le p_{jk} < a_j \\ p_{jk} & if \quad a_j \le p_{jk} \le b_j \\ p_{jk} & if \quad b_j < p_{jk} \le b'_j \\ \infty & if \quad p_{jk} > b'_j \end{cases} \tag{4}$$

In equations (3) and (4), p_{0k} implies arrival time of vehicle $k \in K$ at terminal depot after serving all customers in the route, and a'_0, b'_0, and s_{0k} are all not defined. The objective function as well as all corresponding constraints are formulated as follows:

$$\min \quad \sum_{k \in K} \sum_{(i,j) \in A} c'_{ij} x_{ijk} \tag{5}$$

s.t.

$$\sum_{k \in K} \sum_{j \in V} x_{ijk} = 1 \qquad \forall i \in C \tag{6}$$

$$\sum_{j \in C} x_{0jk} = 1 \qquad \forall k \in K \tag{7}$$

$$\sum_{i \in C} x_{i0k} = 1 \qquad \forall k \in K \tag{8}$$

$$\sum_{i \in V} x_{ihk} = \sum_{j \in V} x_{hjk} \qquad \forall h \in C, \forall k \in K \tag{9}$$

$$s_{ik} + t_{ij} \le (1 - x_{ijk}) M + s_{jk} \qquad \forall i \in C, \forall j \in C, \forall k \in K \tag{10}$$

$$y_{ijk} \le x_{ijk} Q \qquad \forall (i,j) \in A, \forall k \in K \tag{11}$$

$$\sum_{j \in C} y_{0jk} = \sum_{j \in C} d_j \sum_{i \in V} x_{ijk} \qquad \forall k \in K \tag{12}$$

$$\sum_{i \in V} y_{ijk} + (u_j - d_j) \sum_{i \in V} x_{ijk} = \sum_{i \in V} y_{jik} \qquad \forall j \in C, \forall k \in K \tag{13}$$

$$a_i \le s_{ik} \le b'_i \qquad \forall i \in C, \forall k \in K \tag{14}$$

$$a_0 \le l_k \qquad \forall k \in K \tag{15}$$

$$x_{ijk} \in \{0,1\} \qquad \forall (i,j) \in A, \forall k \in K \tag{16}$$

$$y_{ijk} \in Z^+ \qquad \forall (i,j) \in A, \forall k \in K \tag{17}$$

The objective function (5) is to minimize overall costs of routes including fixed vehicle costs, travel costs, and penalties. Constraints (6) are assignment constraints. Every customer is visited exactly once. Constraints (7)-(9) are flow conservation constraints. They imply that all routes must originate and terminate at depot. Also if a vehicle travels to an intermediate vertex $h \in C$, the vehicle should leave from it. Constraints (10) state that if arc $(i,j) \in A$ is traversed by vehicle $k \in K$, the service at vertex $j \in C$ cannot start earlier than the service at vertex $i \in C$; M is referred to a large value. Constraints (11) impose limited vehicle capacity on all arc $(i,j) \in A$. Constraints (12) indicate load of vehicle $k \in K$ before leaving depot, which combines all delivery requests in the route. Constraints (13) are loading flow of vehicle $k \in K$ after serving vertex $j \in C$. Constraints (14) define the feasibility condition of service start time that must be within $[a_i, b'_i]$. Note that if a vehicle $k \in K$ arrives at vertex $i \in C$ before a_i (but not earlier than a'_i), the service cannot start immediately, the vehicle has to wait until a_i. Constraints (15) define departure time from depot of vehicle $k \in K$; due to early arrival penalty, vehicle does not necessarily depart at a_0. Constraints (16) are binary requirements on arc flow variables x_{ijk}, while constraints (17) are integer requirements on load variables y_{ijk}.

4. BRANCH-AND-PRICE EXACT-BASED APPROACH

An optimization-based branch-and-price procedure comprises column generation embedded in branch-and-bound algorithm (Lubbecke and Desrosiers, 2005; Villeneuve *et al*., 2005). The procedure begins with the first node (called root node) of the branch-and-bound tree. The VRPSTWSPD along with its initial solutions are solved using column generation until optimal solution is found. If the solution obtained is fractional, branching scheme is required. Two child-nodes are then created, each of which is assigned an additional bounding constraint, and column generation is called again. The procedure continually branches until the obtained solution is integer. Then, the next untreated node is selected to process. The procedure terminates when all nodes are treated and the branch-and-bound tree cannot further be branched. In our branching strategy, the number of vehicles is taken as the first branching decision if the solution contains a fractional number of vehicles. Branching on arc flow variables is then taken if the solution is still a non-integer solution. For more details of branching scheme as well as the structure of column generation, interesting reader is referred to Qureshi *et al*. (2009).

Applying column generation to the VRPSTWSPD decomposes the problem into a set partitioning master problem (MP) and a newly developed pricing subproblem (SP). The MP aims at selecting a set of routes (or columns) to cover all customer demands at minimum cost, while the SP finds at every iteration non-dominated feasible routes with negative reduced costs and feeds such routes to the MP. Since only a set of necessarily feasible routes is treated in every column generation iteration, the MP is therefore reduced to a restricted master problem (RMP). Let P be a set of feasible routes, c_p be cost of route $p \in P$, e_p be binary variable equal to one if route $p \in P$ is selected in the solution and zero otherwise, and a_{ip} be the number of times route $p \in P$ visits vertex $i \in C$. However, if elementary requirement goes into effect, a_{ip} will then become binary. The RMP can mathematically be described as follows:

$$\min \quad \sum_{p \in P} c_p e_p \tag{18}$$

$$\text{s.t.} \quad \sum_{p \in P} a_{ip} e_p = 1 \qquad \forall i \in C \tag{19}$$

$$e_p \in \{0,1\} \qquad \forall p \in P \tag{20}$$

To initiate column generation, $|C|$ columns, each representing a single customer route, are fed to the RMP; then linear programming relaxation of the RMP is solved by simplex algorithm. Meanwhile, dual variables $\mu_i, \forall i \in C$, which are by-products from constraints (19), are also obtained. These dual variables are inevitable for the SP to calculate the reduced cost of route by subtracting from its original cost. Since every route must start and end at depot, dual variable of depot (μ_0) is also defined. However, there is no initial constraint added in the RMP for the depot, which signifies the restriction on number of vehicles used. Hence, μ_0 is initially set to zero and might further be computed based on additional constraints for the depot, i.e. when branching on the fractional number of vehicles. Column generation works iteratively between the RMP and the SP until no negative reduced cost route can be found in the SP.

At every column generation iteration, the current solution of linear relaxation of the RMP (z_{RMP}) yields an upper bound (z_{UB}) on the optimal value. While a valid lower bound (z_{LB}) is calculated using $z_{LB} = z_{RMP} + z^*_{SP}$, where z^*_{SP} is the optimal value of the SP, i.e. the most

negative reduced cost route. When column generation achieves optimality at each node of the branch-and-bound tree, i.e. no negative reduced cost route can be found by the SP, z_{RMP} of the current RMP fundamentally becomes z_{LB} (Chabrier, 2006). If the current optimal z_{RMP} contains integer solution, it will be set as the new global upper bound (z_{GUB}) in case it is less than the previous one. After entire nodes of the branch-and-bound tree are searched, the current z_{GUB} represents the optimal solution.

4.1 A Two-Stage Pricing Subproblem Solution Technique

The feasibilities of routes generated in the SP rely on constraints (7)-(17). When generating such routes, some complexities might particularly be taken into consideration in the SP including 1) Penalty function is not a non-decreasing function, 2) Possible arrival time at each customer is restrained by definitions of maximum early arrival time and late arrival time, 3) Vehicle capacity along its route is not monotonous, and 4) Repetition of customers visited in the same route is not allowed (elementary requirement). Therefore, to tackle the SP, a two-stage solution technique is then developed. More specifically, a set of candidate feasible routes is generated in the first stage without considering optimal departure times from depot, which are left to be found in the second stage. An optimal departure time is the starting time of the vehicle to leave depot in which overall costs of the route are minimized. In the following, a two-stage solution technique for the novel SP is briefly explained.

4.1.1 First stage: generating a set of candidate routes

In the first stage, a labeling algorithm, which is an extended version of Dell'Amico *et al*. (2006), Fagerholt (2001), and Feillet *et al*. (2004), is employed to generate a set of candidate routes with negative reduced costs. The key feature in this stage is that violating customer's hard time windows is still possible within $[a'_i, b'_i]$, yet penalty is temporarily not considered. Indeed, a vehicle needs waiting at no cost if it arrives early and the delayed service can also occur without penalty. The reduced cost of the route is the sum of the reduced costs of all arcs traversed in the route. The reduced cost $\bar{c}_{ij}$ of arc $(i,j) \in A$ disregarding penalty is computed as follows:

$$\bar{c}_{ij} = c_{ij} - \mu_i \qquad \forall i \in V \tag{21}$$

In the process, a partial path m is characterized by a label L_i^m, which contains useful components such as $last(L_i^m)$: current visited vertex $i \in V$, $\bar{c}(L_i^m)$: the reduced cost of the path m starting from depot, $dep(L_i^m)$: departure time from depot, $pre(L_i^m)$: predecessor vertex, $no(L_i^m)$: label number, $\tau(L_i^m)$: time resource, $q(L_i^m)$: load resource, $diff(L_i^m)$: accumulated load-difference resource, $wt(L_i^m)$: accumulated waiting time, $dt(L_i^m)$: accumulated delayed time, and $v_h(L_i^m), \forall h \in V$: unreachable resources considering at vertex $i \in V$. Unreachable resources, which consist of $|V|$ extra resources, help to forbid repetition of vertex visited in the path to satisfy elementary requirement. Each unreachable resource can be defined either reachable (taking value zero) or temporarily unreachable (0.5) or unreachable (one). The definitions of reachable, temporarily unreachable, and unreachable resources will be given later.

At initialization, the partial path m is constructed at depot, and it is labeled by L_0^m. As the

optimal departure time from depot will be determined in the second stage, an "as early as possible" departure time is instead considered for the path to particularly ensure that any possible departure time will not be neglected. Components $dep(L_0^m)$ and $\tau(L_0^m)$ are thus set to the earliest possible departure time, i.e. max(a_0, a_r-t_{0r}), where $r \in C$ is the prospective first customer of the path m. Please be noted that waiting at first customer of the path is never allowed. Other components including $\bar{c}(L_0^m)$, $q(L_0^m)$, $diff(L_0^m)$, $wt(L_0^m)$, and $dt(L_0^m)$ are all set to zero. The unreachable resource $v_0(L_0^m)$ is initiated as one due to already visited; the remaining unreachable resources $v_h(L_0^m), \forall h \in C$ can take values either zero or 0.5 or one depending on feasibility conditions, which will be described later.

Labeling algorithm is executed by extending partial paths from their current vertices to all possible successors. Note that the terminating depot (artificial vertex), which all partial paths must return to in order to complete their extensions (i.e. a complete path with negative reduced cost represents a candidate route $p \in P$), is always a possible successor for any vertex on any partial path at any time. The feasibility criteria of path extension in the labeling algorithm must assure that some possible schedules are not excluded as it could significantly impact on an absence of the optimal solution. Given the partial path m, its extension from current vertex $i \in V$ to successor vertex $j \in C$ is considered feasible if all equations (22)-(25) are satisfied.

$$b_i' + t_{ij} \geq a_j' \tag{22}$$

$$\tau(L_i^m) + t_{ij} \leq b_j' \tag{23}$$

$$q(L_i^m) + d_j + \max\{0, (u_j - d_j) + diff(L_i^m)\} \leq Q \tag{24}$$

$$v_j(L_i^m) < 1 \tag{25}$$

Focusing on equation (22), it is particularly developed instead of usual inequality $\tau(L_i^m) + t_{ij} \geq a_j'$. This is to retain all possible schedules at customers for further processing in the seconds stage as explained. Since all paths start from the depot at the "as early as possible" departure times, hence time resources at all vertices along the paths are the least ones, which are not necessarily the optimal ones owing to early arrival penalties. Due to the definition of maximum possible arrival time at vertex $i \in C$ within $[a'_i, b'_i]$ exists, use of inequality $\tau(L_i^m) + t_{ij} \geq a_j'$ might lead to an absence of some significant extensions. Instead, use of equation (22) attempts to keep all possible schedules for further processing in the second stage and to prevent the absence of the optimal solution. Once again, the optimal departure time from depot would be found in the second stage.

Equation (24) is also a particular criterion for the VRPSTWSPD subproblem. The vehicle capacity along its route is becoming non-monotonous as both pickup and delivery demands are simultaneously served at each visit, and such fluctuated capacity also plays a significant role in path extensions. For better understanding on this issue, an illustrative example is shown in Figure 2. Given that a problem considered consists of one depot, two customers with known demands, and a single vehicle with fixed capacity of 100. The resulting routing plans of this problem can be displayed in two cases. Both cases seem to be feasible but case 1 is, in fact, not a feasible plan. Once the vehicle leaves depot, total loads of the vehicle equal 100, and visiting customer no. 1 prior to customer no. 2 results definitely in an excess vehicle capacity. Use of equation (24) attempts to prevent such path extension of case 1.

Case 1

Case 2

Figure 2. Examples of routing plan

Every time the partial path m is successfully extended from current vertex $i \in V$ to its successor vertex $j \in C$, the associated label is changed to L_j^m. Components of L_j^m must also be updated as follows:

$$\tau(L_j^m) = \max\{\tau(L_i^m) + t_{ij}, a_j\} \tag{26}$$

$$\bar{c}(L_j^m) = \bar{c}(L_i^m) + \bar{c}_{ij} \tag{27}$$

$$wt(L_j^m) = wt(L_i^m) + \max\{0, a_j - (\tau(L_i^m) + t_{ij})\} \tag{28}$$

$$dt(L_j^m) = dt(L_i^m) + \max\{0, (\tau(L_i^m) + t_{ij}) - b_j\} \tag{29}$$

$$q(L_j^m) = q(L_i^m) + d_j + \max\{0, diff(L_i^m) + (u_j - d_j)\} \tag{30}$$

$$diff(L_j^m) = \min\{0, diff(L_i^m) + (u_j - d_j)\} \tag{31}$$

At some vertex $j \in C$, if factor $diff(L_i^m) + (u_j - d_j)$ based on equation (30) gives positive value, i.e. due to accumulated pickup demands are currently more than accumulated delivery demands, extra loads must be taken into account in load resource $q(L_j^m)$. That is to say, the vehicle needs to leave some spaces for such extra loads. After that, the accumulated load-difference resource of vertex $j \in C$, $diff(L_j^m)$, is set to zero again before further processing as in equation (31).

To update unreachable resources at vertex $j \in C$, $v_j(L_j^m)$ becomes one due to already visited. Some other vertex $h \in C$ would be unreachable ($v_h(L_j^m)$=1) if either it has already been visited or its extension from $j \in C$ to $h \in C$ does not satisfy equations (22)-(24). In particular, if extension from $j \in C$ to $h \in C$ does not just satisfy only equation (22), it is then defined as temporarily unreachable ($v_h(L_j^m)$=0.5). As time resource is a non-decreasing resource, it might be possible to include vertex $h \in C$ to the partial path m afterwards. In all other cases, vertex $h \in C$ would definitely be reached from vertex $j \in C$ ($v_h(L_j^m)$=0).

When two or more partial paths lead to the same current visited vertex, they must be compared and evaluated. All dominated partial paths are eliminated and no longer considered for extensions. This strategy is very useful to reduce storage space and computational time of labeling algorithm. The dominance criteria can be summarized as follows. Given that two

partial paths m and n end at vertex $i \in C$, labels associated to these partial paths are L_i^m and L_i^n, respectively. The partial path m dominates the partial path n if and only if $\bar{c}(L_i^m) \le \bar{c}(L_i^n)$, $\tau(L_i^m) \le \tau(L_i^n)$, $q(L_i^m) \le q(L_i^n)$, $wt(L_i^m) \le wt(L_i^n)$, $dt(L_i^m) \le dt(L_i^n)$, $v_h(L_i^m) \le v_h(L_i^n), \forall h \in V$, and at least one of these inequalities is strict. As the reduced cost calculation in this stage is not an actual calculation, i.e. all penalties are disregarded, two additional inequalities, i.e. $wt(L_i^m) \le wt(L_i^n)$ and $dt(L_i^m) \le dt(L_i^n)$, are then requisite in the dominance rules to precisely evaluate the partial paths.

4.1.2 Second stage: finding the optimal departure times of all candidate routes

Each candidate route obtained in the first stage must be processed to find an optimal departure time of the route. In this stage, all penalties concerned are now active. Given a candidate route $p \in P$ and the first customer $r \in C$ of the route, all possible departure times in range $[(\max(a_0, a_r - t_{0r})), (b'_r - t_{0r})]$ must be taken into consideration. As time windows and travel times are both integer, size of possible departure times that need to be considered is $|(b'_r - t_{0r}) - (\max(a_0, a_r - t_{0r}))|$.

The route p begins at depot; setting initial routing cost to zero, and time resource to the earliest departure time, i.e. $\max(a_0, a_r - t_{0r})$. Next, the vehicle leaves the depot to visit a given set of customers (all sequences were already fixed from the previous stage). Every time the vehicle traverses between two points, cost and time resource of the route p must be updated. The updates of cost and time resource follow equation (3) and equation (26), respectively. After serving all customers, the vehicle returns to the terminal depot. Then, the procedure starts again by setting initial cost of the route p to zero, and time resource to the next possible departure time. All corresponding departure times are proceeded in increasing order. If some possible departure time cannot satisfy soft time window at some vertex $i \in C$ due to arriving before a'_i, this departure time and its calculations will then be eliminated, and the procedure will start again with the next possible departure time. On the contrary, if some possible departure time results in arriving later than b'_i at some vertex $i \in C$, the procedure on the route p will immediately terminate, and all remaining departure times will not be considered. After all, the departure time that leads to cost minimization is selected. If two or more departure times lead to the same cost, the one with minimum waiting time is taken as one of our objectives. Eventually, all candidate routes must be optimally processed to determine their departure times and their corresponding overall costs before feeding them to the RMP.

5. GENETIC ALGORITHM

Genetic algorithm (GA), which is one of effective metaheuristics, has been widely applied to combinatorial optimization problems such as the VRP. Interesting reader is referred to Braysy and Gendreau (2005) and Caserta and Vob (2010) for comprehensive survey of metaheuristics. The GA works for a number of generations (iterations) in which characteristics of individuals (or solutions) in the current population are formed and placed to the new population via several genetic operators.

At each generation of the proposed GA, a two-stage operation is performed. In the first stage, all routes are primarily generated without considering optimal departure times from depot, i.e. considering to depart as early as possible. Furthermore, violating customer's hard time windows within $[a'_i, b'_i]$ is feasible without considering penalty. Later, in the second stage, after executing all genetic operators, each individual in the current population is then decoded

and each route in the individual is optimally processed to determine its departure time from depot using the same procedure as in the second stage of the pricing subproblem algorithm. Subsequently, the corresponding overall costs of all routes in each individual including fixed vehicle costs, travel costs, and penalties are recalculated, and an inverse of such overall costs provides a fitness value of the individual. It is worth pointing out that the two-stage operation in the proposed GA is implicitly similar to the two-stage subproblem algorithm of the branch-and-price approach, where a set of candidate routes is generated in metaheuristic fashion instead of exact fashion.

After termination of the proposed GA, the best individual found during entire search is finally taken, representing the (nearly) optimal solution. The proposed GA and its operators can be summarized as follows.

5.1 Individual Representation

An individual in the population is encoded by a two-row representation, where the first row represents an integer string (called chromosome) of length $|C|$, and the second row indicates a set of assigned vehicles. As displayed in Figure 3, an illustrative instance with 5 customers is routed. Each integer (called gene) in the chromosome represents a customer node; depot is always omitted. In this manner, after leaving depot, vehicle 1, respectively, visits customer 5 and 1 before returning to depot. Vehicle 2 visits depot, customer 2, 3, and depot in sequence, while vehicle 3 is a single customer route of customer 4.

Chromosome	5	1	2	3	4
Vehicle reference	1	1	2	2	3

Figure 3. Solution representation

5.2 Construction Algorithm

To start the proposed GA, an initial population of D individuals must be created (D is fixed to 100 individuals in this study). First half of the initial population is obtained using an extended insertion heuristic with residual capacity criterion (ψ_{RC}) of Dethloff (2001), where the algorithm must incorporate additional soft time window constraints. While the second half is constructed from an effective greedy algorithm with reinsertion of single customer routes. Every time a customer is added to (or removed from) the existing route, feasibilities on capacity constraints and soft time window constraints must be checked. The extended insertion-based heuristic generally gives better quality of individuals, while the greedy algorithm results in more diversity.

In particular, for more advanced exploration of the solution space, the proposed GA considers a new search strategy over all generations. Such strategy is to allow generating intermediate infeasible routes with taking value M into account. An intermediate infeasible route is a route that violates soft time window constraint at some vertex $i \in C$ due to arriving earlier than a'_i. Given that a vehicle $k \in K$ leaves vertex $h \in V$ to vertex $i \in C$, which causes intermediate infeasible arrival, i.e. $a'_i - p_{ik} > 0$, the travel cost involved is calculated by $c_{hi} + M(a'_i - p_{ik})$.

5.3 Selection Operator

In this operator, individuals in the current population are selected to be parent (father and

mother) individuals. A tournament selection (Ghoseiri and Ghannadpour, 2010) is employed for this purpose. Two copies of the population are made. In the first copy, all individuals are ranked in order of decreasing their fitness values. Two consecutive individuals are compared and the one with higher fitness value is chosen to be father individual. In the second copy, all individuals are arbitrarily ranked and processed similarly. Finally, a set of father individuals with size $|D/2|$ and a set of mother individuals with size $|D/2|$ are ready for mating.

5.4 Crossover Operator (recombination, with a rate of 80%)

Characteristics of father and mother chromosomes are shared and carried on into offspring individuals during crossover. The best cost route crossover (BCRC) of Ombuki *et al*. (2006) is called to create two offspring from a pair of parents. Given that a father individual (F) and a mother individual (M) are selected for mating. To create the first offspring, one route in F is randomly chosen, all customers appeared in the selected route of F are then removed from M. Next, all removed customers in M are, one-by-one randomly, reinserted to the existing routes of M in their best possible locations. Note that some removed customer might be reinserted to a new route if possible insertion to all existing routes of M cannot be found. To create the second offspring, a route in M is in turn randomly selected and the same procedure is called. Finally, a set of D offspring individuals is thus obtained after crossover and they will become candidates for the new population.

5.5 Mutation Operator (with a rate of 20%)

Mutation operator attempts to prevent too early convergence of the solution by enlarging the search region. Two mutation strategies of Alvarenga *et al*. (2007) are applied to a randomly selected subset of offspring individuals. Both mutation strategies are equally likely to be applied. The first strategy is a similar customer exchange. In each selected offspring individual, the operator randomly selects a route and a vertex $i \in C$ associated to the route. Then, the operator searches for a vertex $j \in C$ located in other routes in which the difference of these two customers, i.e. $|a_i - a_j|$, is minimal, and tries to exchange their locations. The second strategy is a random customer migration, where a randomly chosen vertex $i \in C$ from one route is randomly migrated to other existing route.

5.6 Intensification (with a rate of 30%)

After mutation, a fixed number of offspring individuals is randomly selected. A 2-opt* local search of Potvin and Rousseau (1995) is then applied to improve the quality of these offspring.

5.7 Fitness Value

The fitness value must be measured at the end of every generation in order to evaluate the performance of all individuals generated in the current population, which further influences creation of the next population. The evolution of the GA relies on these fitness values. The fitness value of each individual equals the inverse of overall costs of all corresponding routes in the individual as described previously.

5.8 Replacement and Elitism

A set of *D* offspring generated through genetic operators explained above is eventually set to be the individuals of the new population. The generations of GA are continually repeated until terminating conditions are met. The proposed GA terminates when either 200 iterations are run or 20 consecutive populations result in similar individuals. To preserve the quality of solution over generations, 2% of the best individuals (highest fitness values) found in the current population are kept and brought directly to the new population.

6. RESULTS AND DISCUSSIONS

Our numerical experiments were conducted on an Intel Core i7 CPU, 2.67 GHz processor, and 4 GB of memory, and all codes were implemented in MATLAB (R2009a). Test instances for evaluating the proposed solution algorithms are composed of Solomon's VRPTW benchmark set (Solomon, 1987). For evaluating the proposed GA, 10 experiments were performed on each instance and the one with best solution produced was selected.

In Solomon's benchmark, six different types of problem are contained; however, only the RC1 type was picked for testing. Generally, the RC1 type consists of eight different instances. All information including coordinates of vertices, vehicle capacity, service time, time window of depot, and customer's hard time windows are provided in each instance. A single depot is located at the center of a two-dimensional 100*100 Euclidean space, while customer's locations are geographically dispersed in a form of mixed cluster and randomness. All instances of the RC1 type have the same coordinates of vertices, whereas they differ from each other in characteristics of customer's hard time windows. Figure 4 displays an instance of the RC1 type, named RC101.

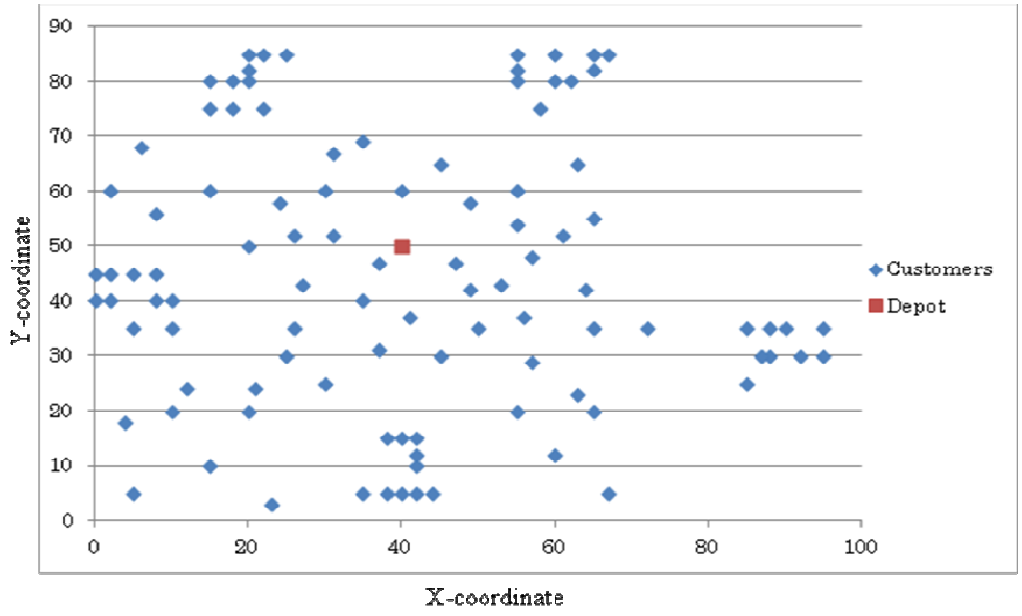

Figure 4. Scattered points of RC1 type (RC101 instance)

In particular, delivery demands and pickup demands of all customers were randomly generated between the interval [0,90]. Note that each complete instance comprises 100 customers, yet smaller sizes of the concerned instance could be considered by taking the first few customers such as 25, 40 or 75 customers. Travel cost on arc connected two points was measured using Euclidian distance calculation and was rounded down to nearest integer if

fractional. Travel time on arc was set equal to the sum of travel cost on the arc and service time at the starting point. Following data reported in Qureshi *et al*. (2009), vehicle operating cost (VOC) was set to 14.02 yen/minute and fixed vehicle cost was taken as 10417.5 yen/vehicle. Both values were gathered from surveys of Japanese logistics firms. For the sake of simple calculations, however, the VOC was converted to as one unit of travel cost equals one unit of travel time (i.e. VOC=1). Fixed vehicle cost was consequently scaled to the same level, i.e. 743 unit cost/vehicle. Early arrival penalty and late arrival penalty were set to three times and five times of the VOC, respectively. Due to arbitrary limits and simplicity of problems, a soft time window at vertex $i \in C$, i.e. $[a'_i,b'_i]$ was particularly extended up to 10 units from the given hard time window $[a_i,b_i]$ instead of using maximum a'_i and b'_i based on Eq.(1)-(2).

Table 1 and 2 provide numerical results of the branch-and-price algorithm and the proposed GA, respectively, for solving benchmark instances. In Table 1, the most-left columns show instance name with size of customers considered. Column "BB" represents number of nodes explored in the search tree. Column "LB" gives lower bound value obtained when column generation ended at the root node. Column "Z" is the optimal cost, while columns "WC" and "DC" specify, respectively, early arrival penalties and late arrival penalties. Column "K" gives the number of vehicles needed in the optimal solution. Column "Run" shows total number of iterations required to reach the optimal solution. Column "Label" shows an average number of labels generated at the root node. Column "CPU" reports computational time (in second) required starting from the root node to reach the optimal solution.

Table 1. Summary of exact solutions using the branch-and-price approach

Instance		BB	LB	Z	WC	DC	K	Run	Label	CPU
Name	Size									(s)
RC101	25	12	3360.4	3505	0	20	4	85	4279	47.2
RC102	25	11	3740.2	4213	0	0	5	101	16779	108.7
RC103	25	15	3609.2	4208	0	0	5	154	45261	334.3
RC104	25	14	2974.8	3379[a]	0	0	4	240	141801	1279.7
RC105	25	8	3947.1	4261	0	0	5	73	9589	69.2
RC106	25	15	4123.9	4258	0	5	5	88	13056	60.8
RC107	25	21	3849.2	4203[a]	0	0	5	106	41122	198.2
RC108	25	25	4375.6	4998[a]	0	0	6	145	37320	225.6
RC101	40	15	5893.8	6155	0	40	7	109	6462	129.5
RC102	40	17	5785.4	6056[a]	0	0	7	190	27760	664.4
RC103	40	15	5557.5	6045[a]	0	0	7	166	79128	1327.1
RC104	40	28	4727.9	5266[a]	0	5	6	366	299453	3962.4
RC105	40	11	6245.0	6847	6	0	8	121	14460	273.2
RC106	40	20	6479.4	6824[a]	0	5	8	146	18377	294.1
RC107	40	23	6312.4	6768[a]	0	0	8	124	54263	351.4
RC108	40	20	6363.9	6759[a]	0	0	8	163	112220	848.5

[a] The best integer solution found after 10 hours running of the branch-and-price approach.

The separated details of construction algorithm and the proposed GA are given in Table 2. Column "Total CPU" combines both CPU of construction algorithm and the GA. Column "Δt" describes the percentage decrease in computational time between exact solution and approximate solution, while "Δz" shows percentage gap between the optimal cost and the

best cost found by the proposed GA. The terminology "N/A" informs that a comparison of exact solution and approximate solution is not available as size of instance is too large for the branch-and-price approach, thus neither best integer solution nor lower bound obtained at the root node is found.

Table 2. Summary of approximate solutions using the proposed GA

Instance		Construction algorithm			GA			Total CPU	Δt	Δz
Name	Size	Z	K	CPU(s)	Z	K	CPU(s)	(s)	(%)	(%)
RC101	25	4370	5	4.6	3505	4	32.7	37.3	-21.0	0
RC102	25	4273	5	4.3	4213	5	30.5	34.8	-68.0	0
RC103	25	4291	5	5.2	4208	5	50.5	55.7	-83.3	0
RC104	25	3409	4	6.9	3379	4	73.3	80.2	-93.7	0
RC105	25	4689	5	4.7	4264	5	56.8	61.5	-11.1	+0.1
RC106	25	4997	6	4.5	4258	5	48.0	52.5	-13.7	0
RC107	25	4267	5	4.3	4203	5	46.5	50.8	-74.4	0
RC108	25	5027	6	4.3	5000	6	45.4	49.7	-78.0	+0.04
RC101	40	7162	8	10.5	6177	7	116.6	127.1	-1.9	+0.4
RC102	40	7049	8	9.7	6057	7	86.1	95.8	-85.6	+0.02
RC103	40	6930	8	12.4	6029	7	189.3	201.7	-84.8	-0.3[b]
RC104	40	5235	6	13.3	5171	6	138.8	152.1	-96.2	-1.8[b]
RC105	40	7922	9	12.6	6870	8	136.7	149.3	-45.4	+0.3
RC106	40	7152	8	9.3	6817	8	125.6	134.9	-54.1	-0.1[b]
RC107	40	6865	8	9.0	6752	8	148.4	157.4	-55.2	-0.2[b]
RC108	40	6886	8	11.4	6757	8	108.3	119.7	-85.9	-0.03[b]
RC101	75	13437	15	30.3	12120	14	276.2	306.5	N/A[c]	N/A[c]
RC102	75	12460	14	29.2	11242	13	231.2	260.4	N/A[c]	N/A[c]
RC103	75	12302	14	27.4	11045	13	369.8	397.2	N/A[c]	N/A[c]
RC104	75	10406	12	33.7	10109	12	805.8	839.5	N/A[c]	N/A[c]
RC105	75	13302	15	35.2	12006	14	514.1	549.3	N/A[c]	N/A[c]
RC106	75	13032	15	31.4	12106	14	326.9	358.3	N/A[c]	N/A[c]
RC107	75	12870	15	26.7	11782	14	596.4	623.1	N/A[c]	N/A[c]
RC108	75	11116	13	33.8	10157	12	857.3	891.1	N/A[c]	N/A[c]

[b] The percentage gap gives negative value, i.e. the solution cost of the GA is less than that of the branch-and-price approach since the branch-and-price approach fails to solve the instance to optimality within the setting time limit. The solution reported is just the best solution found after 10 hours of computational time.

[c] No comparison between exact solution and approximate solution is available as neither the best integer solution nor lower bound value could be produced by the branch-and-price approach.

Both the branch-and-price approach and the proposed GA were effective when performing on instances with 25-customer. The branch-and-price approach was able to solve almost all of the small-sized instances to optimality at reasonable computational time. For RC104, RC107, and RC108 with 25-customer, which could not optimally be solved, the solutions reported are just the best integer solutions found after 10 hours of computational time, i.e. not the optimal ones. In comparison to the exact solutions, the proposed GA could produce the solutions with the same number of vehicles used, while the solution costs obtained by the proposed GA of some instances were slightly high such as in RC105 (0.1%),

and RC108 (0.04%). Use of the proposed GA resulted in the decrease of computational time for all of the small-sized instances.

When involving instances with larger size and/or more complex, the branch-and-price approach often failed to solve them to optimality within the setting time limit of 10 hours of computational time. Due to the facts that a huge number of labels are generated in the SP, more column generation iterations are required, and a large number of nodes in the branch-and-bound tree are to be explored, the exact approach typically spent much longer computational time. Only RC101 and RC105 with 40-customer instances were solved to optimality. Both algorithms still gave the same number of vehicles used. Note that for some instances with 40-customer such as RC103, RC104, RC106, RC107, and RC108, even the best integer solutions produced by the branch-and-price approach within 10 hours of computational time were worse than those obtained by the proposed GA due to the complexity of problems.

For all instances with 75-customer, neither optimality nor lower bound nor the best integer solution obtained at the root node of the search tree could be acquired by use of the branch-and-price approach. Therefore, the results produced by the proposed GA are just given in Table 2 without any comparison. However, based on the performance of the proposed GA on the small-sized and medium-sized instances, it is expected that these results performed on larger-sized instances are also close to the optimal ones.

7. CONCLUSIONS

This paper has introduced the VRPSTWSPD, which represents a class of distribution and routing systems of convenience store industry. The VRPSTWSPD theoretically combines characteristics of two existing problems in the VRP family, i.e. the VRPSTW and the VRPSPD. The 3-index model formulation of the VRPSTWSPD has been first developed in this paper, taking into account several relevant constraints such as soft time window constraints and capacity constraints. Branch-and-price exact-based approach has been proposed to optimally solve the VRPSTWSPD. Decomposition of the VRPSTWSPD has led to the MP and the newly developed SP. Complexities of the SP rely on all of the following: 1) non-decreasing functions of cost and penalty, 2) non-monotonic vehicle capacity and 3) elementary requirement. To cope with the SP, the two-stage subproblem solution algorithm has also been proposed to generate a set of candidate routes in exact environment.

As NP-hard the VRPSTWSPD is, an efficient GA metaheuristic-based approach has been developed. The proposed GA considers an advanced search strategy to expand the exploration of solution space. Such strategy allows generating intermediate infeasible solutions with high penalties. Various genetic operators have been presented in order to enhance the performance of the proposed GA. The exact approach is superior to the metaheuristic approach due to the fact that only the former can give lower bounds. On the other hand, the GA has advantage over the exact approach when tackling large-scale network, where its usage would absolutely provide good solution to the problem in reasonable computational time. Comparisons of these two solution algorithms on small-sized benchmark instances signify that the proposed GA is remarkably efficient. The results indicate that the proposed GA worked well with minor tolerance. It is worth mentioning that the exact approach is extremely important although it is quite respected only to small-sized instances. Use of the exact solution as reference to assess performance of the proposed GA offers the high confidence level to apply the proposed GA to handle real-life instances, which are considerably larger and more complex.

REFERENCES

Ai, T.J., Kachitvichyanukul, V. (2009) A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 36, 1693-1702.

Alvarenga, G.B., Mateus, G.R., de Tomi, G. (2007) A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34, 1561-1584.

Ando, N., Taniguchi, E. (2006) Travel time reliability in vehicle routing and scheduling with time windows. *Networks and Spatial Economics*, 6, 293-311.

Baldacci, R., Mingozzi, A., Roberti, R. (2012) Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218, 1-6.

Balakrishnan, N. (1993) Simple heuristics for the vehicle routing problem with soft time windows. *Journal of the Operational Research Society*, 44(3), 279-287.

Bianchessi, N., Righini, G. (2007) Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery. *Computers & Operations Research*, 34, 578-594.

Braysy, O., Gendreau, M. (2005) Vehicle routing problem with time windows, part II: Metaheuristics. *Transportation Science*, 39(1), 119-139.

Caserta, M., Vob, S. (2010) Metaheuristics: intelligent problem solving. In Maniezzo, V., Stutzle, T., Vob, S. (eds.), *Matheuristics, Annals of Information Systems*, 10, 1-38. Springer US, ISBN 978-1-4419-1305-0.

Chabrier, A. (2006) Vehicle routing problem with elementary shortest path based column generation. *Computers & Operations Research*, 33, 2972-2990.

Chiang, W.C., Russell, R.A. (2004) A metaheuristic for the vehicle-routing problem with soft time windows. *Journal of the Operational Research Society*, 55, 1298-1310.

Dell' Amico, M., Righini, G., Salani, M. (2006) A Branch-and-Price approach to the vehicle routing problem with simultaneous distribution and collection. *Transportation Science*, 40(2), 235-247.

Dethloff, J. (2001) Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up. *OR Spektrum*, 23, 79-96.

Fagerholt, K. (2001) Ship scheduling with soft time windows: An optimisation based approach. *European Journal of Operational Research*, 131, 559-571.

Feillet, D., Dejax, P., Gendreau, M., Gueguen, C. (2004) An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44, 216-229.

Figliozzi, M.A. (2010) An iterative route construction and improvement algorithm for the vehicle routing problem with soft time windows. *Transportation Research Part C*, 18, 668-679.

Fu, Z., Eglese, R., Li, L.Y.O. (2008) A unified tabu search algorithm for vehicle routing problems with soft time windows. *Journal of the Operational Research Society*, 59, 663-673.

Gajpal, Y., Abad, P. (2009) An ant colony system for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*, 36, 3215-3223.

Gendreau, M., Guertin, F., Potvin, J.Y., Taillard, E. (1999) Parallel tabu search for a real-time vehicle and dispatching. *Transportation Science*, 33(4), 381-390.

Ghoseiri, K., Ghannadpour, S.F. (2010) Multi-objective vehicle routing problem with time

windows using goal programming and genetic algorithm. *Applied Soft Computing*, 10, 1096-1107.

Hashimoto, H., Ibaraki, T., Imahori, S., Yagiura, M. (2006) The vehicle routing problem with flexible time windows and travelling times. *Discrete Applied Mathematics*, 154, 2271-2290.

Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T., Yagiura, M. (2005) Effective local search algorithms for routing and scheduling problems with general time-window constraints. *Transportation Science*, 39(2), 206-232.

Ioachim, I., Gelinas, S., Soumis, M., Desrosiers, J. (1998) A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, 31(3), 193-204.

Ioannou, G., Kritikos, M., Prastacos, G. (2003) A problem generator-solver heuristic for vehicle routing with soft time windows. *Omega; The International Journal of Management Science*, 31, 41-53.

Laporte, G. (2009) Fifty years of vehicle routing. *Transportation Science*, 43(4), 408-416.

Liberatore, F., Righini, G., Salani, M. (2011) A column generation algorithm for the vehicle routing problem with soft time windows. *4OR: A Quarterly Journal of Operations Research*, 9(1), 49-82.

Lubbecke, M.E., Desrosiers, J. (2005) Selected topics in column generation. *Operations Research*, 53(6), 1007-1023.

Min, H. (1989) The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Research Part A*, 23(5), 377-386.

Montane, F.A.T., Galvao, R.D. (2006) A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service. *Computers & Operations Research*, 33, 595-619.

Nagy, G., Salhi, S. (2005) Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries. *European Journal of Operational Research*, 162, 126-141.

Ngaochay, T., Walsh, J. (2011) Paths to success for 7-Eleven in Thailand. *Information Management and Business Review*, 3(1), 1-7.

Ombuki, B., Ross, B.J., Hanshar, F. (2006) Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24, 17-30.

Parragh, S.N., Doerner, K.F., Hartl, R.F. (2008) A survey on pickup and delivery problems: Part 1: Transportation between customers and depot. *Journal für Betriebswirtschaft*, 58(1), 21-51.

Potvin, J.Y., Rousseau, J.M. (1995) An exchange heuristic for routing problems with time windows. *Journal of the Operational Research Society*, 46, 1433-1446.

Qureshi, A.G., Taniguchi, E., Yamada, T. (2009) An exact solution approach for vehicle routing and scheduling problems with soft time windows. *Transportation Research Part E*, 45, 960-977.

Qureshi, A.G., Taniguchi, E., Yamada, T. (2010) Exact solution for vehicle routing problem with semi soft time windows and its application. *Procedia Social and Behavioral Sciences*, 2, 5931-5943.

Qureshi, A.G., Taniguchi, E., Yamada, T. (2012) Exact solution for vehicle routing problem with soft time windows and dynamic travel time. *Asian Transport Studies*, 2(1). 48-63.

Salhi, S., Nagy, G. (1999) A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of the Operational Research Society*, 50, 1034-1042.

Solomon, M.M. (1987) Algorithms for the vehicle routing and scheduling problems with time

window constraints. *Operation Research*, 35(2), 254-265.

Subramanian, A., Cabral, L.D.A.F. (2008) An ILS based heuristic for the vehicle routing problem with simultaneous pickup and delivery and time limit. Proceedings of the 8th European Conference on Evolutionary Computation in Combinatorial Optimization, Naples, Italy, March 26-28.

Subramanian, A., Drummond, L.M.A., Bentes, C., Ochi, L.S., Farias, R. (2010a) A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37, 1899-1911.

Subramanian, A., Ochi, L.S., Uchoa, E. (2010b) New lower bounds for the vehicle routing problem with simultaneous pickup and delivery. In Festa, P. (eds.), *Experimental Algorithms: Lecture Notes in Computer Science*, 6049, 276-287. Springer, Berlin-Heidelberg.

Tagmouti, M., Gendreau, M., Potvin, J.Y. (2007) Arc routing problems with time-dependent service costs. *European Journal of Operational Research*, 181, 30-39.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.Y. (1997) A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31(2), 170-186.

Taniguchi, E., Kakimoto, Y. (2003) Effects of e-commerce on urban distribution and the environment. *Journal of the Eastern Asia Society for Transportation Studies*, 5, 2355-2366.

Taniguchi, E., Shimamoto, H. (2004) Intelligent transportation system based dynamic vehicle routing and scheduling with variable travel times. *Transportation Research Part C*, 12, 235-250.

Taniguchi, E., Thompson, R.G., Yamada, T., Duin, R.V. (2001) *City logistics: Network modelling and intelligent transport systems*. Pergamon, Oxford.

Tasan, A.S., Gen, M. (2012) A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Computers & Industrial Engineering*, 62, 755-761.

The U.S. Agricultural Trade Office Japan, U.S. Embassy, Tokyo (2011) *Japan Retail Food Sector Report 2011*, GAIN Report Number: JA1524. (Online) Available: http://gain.fas.usda.gov/Recent%20GAIN%20Publications/Retail%20Foods_Tokyo%20ATO_Japan_12-21-2011.pdf (accessed November 6, 2012).

Toth, P., Vigo, D. (2002) *The vehicle routing problem: SIAM Monographs on Discrete Mathematics and Applications*, SIAM, Philadelphia.

Villeneuve, D., Desrosiers, J., Lubbecke, M.E., Soumis, F. (2005) On compact formulations for integer programs solved by column generation. *Annals of Operations Research*, 139, 375-388.

Wang, H.F., Chen, Y.Y. (2012) A genetic algorithm for the simultaneous delivery and pickup problems with time window. *Computers & Industrial Engineering*, 62, 84-95.

Westphal, S., Krumke, S.O. (2008) Pruning in column generation for service vehicle dispatching. *Annals of Operations Research*, 159, 355-371.