Journal of the Korean Institute of Industrial Engineers
Vol. 33, No. 1, pp. 33-43, March 2007.

An Optimization Algorithm for The Pickup and Delivery Problem With Time Windows

Jayoung Kang · Hee Jeong Zang · Jangha Kang · Sungsoo Park†

Department of Industrial Engineering, Korea Advanced Institute of Science and Technology

동일경로 제약을 갖는 집배송 차량 경로 수립 문제의 최적화 해법

강자영 · 장희정 · 강장하 · 박성수

한국과학기술원 산업공학과

The pickup and delivery problem with time windows generally involves the construction of optimal routes which satisfy a set of transportation requests under pairing, precedence, time window, vehicle capacity, and availability constraints. In this paper, we added some constraints to the problem and adopted an objective function based on number of used vehicles, total travel distance and total schedule duration to consider more realistic problems. A branch and price algorithm for the problem is proposed and an enumeration method is used for the subproblems. The algorithm was tested on randomly generated instances and computational results were reported.

***Keywords*:** Pickup and Delivery Problem with Time Windows, Branch-and-Price Algorithm

1. Introduction

In the pickup and delivery problem with time windows (PDPTW), an optimal set of routes has to be constructed to satisfy transportation requests. A transportation request is characterized by a pickup location, a delivery location and items to be delivered. Satisfying a transportation request is that one vehicle collects the items at the pickup location and delivers them to the delivery location. Several transportation requests can be served by a vehicle, but items can not be transferred from a vehicle to another.

Since the PDPTW is NP-hard (Desrosiers *et al*., 1995), majority of researches on the PDPTW have focused on heuristic approaches based on tabu search or genetic algorithms. And some researchers constructed benchmark data sets for the PDPTW. Table 1 shows us some recent results on heuristic approaches.

Nanry and Barnes (2000) constructed a set of benchmark instances for the PDPTW based on the Solomon's benchmark data set for the vehicle routing problem with time windows (VRPTW). And their work was extended by Lau and Liang (2001). Li and Lim (2001) also generated a set of benchmark instances from Solomon's one. But, their generating manner is different from the one by Nanry and Barnes (2000). Nanry and Barnes (2000) paired up customer locations in the optimal routes. On the other hand, Li and Lim (2001) randomly paired up the customer locations within routes

This work was supported by the Korea Research Foundation Grant. (KRF-2003-041-D20553)

† Corresponding author : Sungsoo Park, Department of Industrial Engineering, KAIST 373-1 Kusong-dong, Yusong-gu, Daejeon 305-701, Korea
Tel : +82-42-869-3121, Fax : +82-42-869-3110, E-mail : sspark@kaist.ac.kr

Received April 2005; revision received June 2006; accepted July 2006.

Table 1. Recent heuristic approaches for the PDPTW

Authors (Year)	Objective Function (Minimize)	Approach	Benchmark Data Sets
Nanry and Barnes (2000)	total travel time + penalty of violating TW constraints +penalty of violating capacity constraints	reactive tabu search	O
Lau and Liang (2001)	number of vehicles, total travel distance	two phase method	O
Li and Lim (2001)	number of vehicles, total travel distance, total schedule duration and total waiting time	tabu-embedded simulated annealing approach	O
Jih, Kao, and Hsu (2002)	total travel time +total waiting time	family competition genetic algorithm	
Kammarti *et al*. (2004)	total travel distance, total waiting time, total tardiness	hybrid evolutionary approach	
Pankrats (2005)	total travel distance	grouping genetic algorithm	

in solutions obtained by their heuristic approach for the VRPTW. Pankrats (2005) proposed a grouping genetic algorithm and tested it on the benchmark data sets. Since different objective functions were considered in the two papers, Pankrats (2005) simplified the objective function to compare the results. Jih, Kao, and Hsu (2002) suggested a family competition genetic algorithm and it performed better than the genetic algorithm. Kammarti *et al*. (2002) proposed a hybrid approach which was based on genetic operators, tabu search and Pareto dominance method. Even though it was tested on the benchmark instances generated by Li and Lim (2001), the results could not be compared to others because of the difference of the objectives.

In the optimization area, only two algorithms for the PDPTW were presented. Dumas *et al*. (1991) provided a mathematical model and proposed a branch and price algorithm. A dynamic programming algorithm was hired to solve the subproblem. The second optimization algorithm was presented by Savelsbergh and Sol (1998). It is also a branch and price algorithm, but several techniques were employed to improve the performance of the algorithm. Most important techniques were related to branching strategy and algorithms for the subproblem. Their branching strategy focused on assignment decisions instead of routing decisions, and it has more impact on the structure of the solutions than the one proposed by Dumas *et al*. (1991). And they hired polynomial approximation algorithms for the subproblem as long as they could generate columns with negative reduced costs. In order to guarantee the optimality of the solutions, the dynamic programming algorithm was used when the polynomial approximation algorithms did not work. Computational results showed the approximation algorithms saved time a lot. Actually, Savelsbergh and Sol (1998) were interested in the general pickup and delivery problem with Time windows (GPDP) in dynamic environment. In the GPDP, each transportation request may concern multiple pickup locations and delivery locations and visiting order of the locations is not previously determined. (For details, see Savelsbergh and Sol, 1995) But, they assumed that all request specify one pickup location and one or more delivery locations, which have to be visited in a predefined order. Therefore, the problem considered by Savelsbergh and Sol (1998) is not the exact GPDP, but a partially generalized version of the PDPTW. And the algorithm for the problem can be obtained by slightly modifying any algorithm for the PDPTW. That is, the algorithm proposed by Savelsbergh and Sol (1998) is much closer to the algorithm for the PDPTW than to the algorithm for the GPDP.

Even though there are two optimization algorithms for the PDPTW, they have a limit. Both algorithms fix starting time of each vehicle to its earliest starting time and generate solutions which may contain unnecessary waiting time. Since Dumaset *et al*. (1991) and Savelsbergh and Sol (1998) assumed that each vehicle starts its route at its earliest starting time, the algorithms could be valid for their problem. But, this assumption is unreal and also prevents taking various objective functions such as total waiting time and total scheduling duration which were already adopted by several heuristic approaches. In this paper, we consider more realistic problem not allowing this limit and a branch and price algorithm for the problem is presented.

In the next section, characteristics of the problem are described. Section 3 introduces the mathematical model for the new problem. The algorithm is given in section 4 and computational results are provided in section 5. Finally, section 6 gives conclusion.

2. The Pickup and Delivery Problem With Time Windows

The PDPTW is a problem to find an optimal set of routes which satisfy all given transportation requests under the following constraints. A route should end at its starting location, a depot where vehicles are stationed. To satisfy a transportation request, a vehicle collects items at a pickup location and delivers them to a delivery location without any transshipment at an intermediate location. Pairing constraints ensure that a pickup location and a delivery location of each transportation request should be visited by one vehicle. Precedence constraints imply that a vehicle should visit the pickup location before the delivery location of a transportation request. Each location specifies a time window which is defined as a time interval between the earliest arrival time and the latest arrival time. Time window constraints make sure that a service at a location has to be given between the earliest arrival time and the latest arrival time of the location. When a vehicle arrives at a location too early, it is allowed to wait until the earliest arrival time of the location. In this paper, different types of vehicles are considered and each vehicle is characterized by capacity, cost and the depot. Vehicle capacity constraints guarantee that load of items on a vehicle should be less than or equal to the vehicle capacity. We assume that the number of available vehicles for each vehicle type is limited. So availability constraints ensure that the number of used vehicles is less than or equal to the number of available vehicles for each vehicle type.

Besides the above basic constraints, we add four more constraints to the PDPTW. The first one is location capacity constraints. According to the environment of service locations, some big vehicles may not be allowed to enter the locations. For instance, an eight-ton truck can not run on a very narrow alley or hillside slums. Therefore, the location capacity constraints are considered to screen out those infeasible cases. The second one is extra worker constraints. If an item is so heavy that a driver cannot carry it alone, extra workers are needed to deliver the item. And then, we assume that they go the rounds of customers together until the vehicle returns to the depot. Therefore, we consider the extra worker constraints which make sure that the number of extra workers of a route is greater than or equal to the maximum number of necessary extra workers of items to be delivered by the route. The third and the fourth are distance constraints and duration constraints. The duration of a route equals the ending time minus the starting time of a route and it includes travel time, waiting time, and service time. The distance constraints and the duration constraints restrict the travel distance and the schedule duration of each route respectively. The distance and the duration constraints are necessary to improve the working conditions. Without these, an optimal schedule may ask one to work for twelve hours, while it requests the other to work only for three hours. Even though the constraints do not guarantee uniform length of routes, they are helpful for making fair schedules.

From <Table 1>, the objective functions of the PDPTW are related to the number of vehicles, the total travel distance, the total travel time, the total waiting time, or penalty of violation of some constraints. Since we want feasible solutions, we do not consider penalty functions. Dumas *et al*. (1991) used the minimization of total cost, but the cost function was not clearly defined. The primary objective considered by Savelsbergh and Sol (1998) was the minimization of the number of used vehicles. The secondary objective was to minimize the total travel distance. Both objectives can be combined by increasing setup cost of vehicles sufficiently. In this paper, the objective function is the minimization of total cost which is the sum of costs of all routes. The cost of a route consists of vehicle cost, distance cost, and labor cost. The vehicle cost is determined by the vehicle type. The distance cost is directly proportional to the travel distance. The labor cost is the sum of labor costs of the driver and the extra workers. The labor cost of a worker is divided into regular working cost and overtime working cost. We assume that there is a specific time to distinguish regular hours and overtime hours. The regular working cost varies as working hours before the specific time, whereas the overtime working cost varies as working hours after the time. Generally, the overtime working

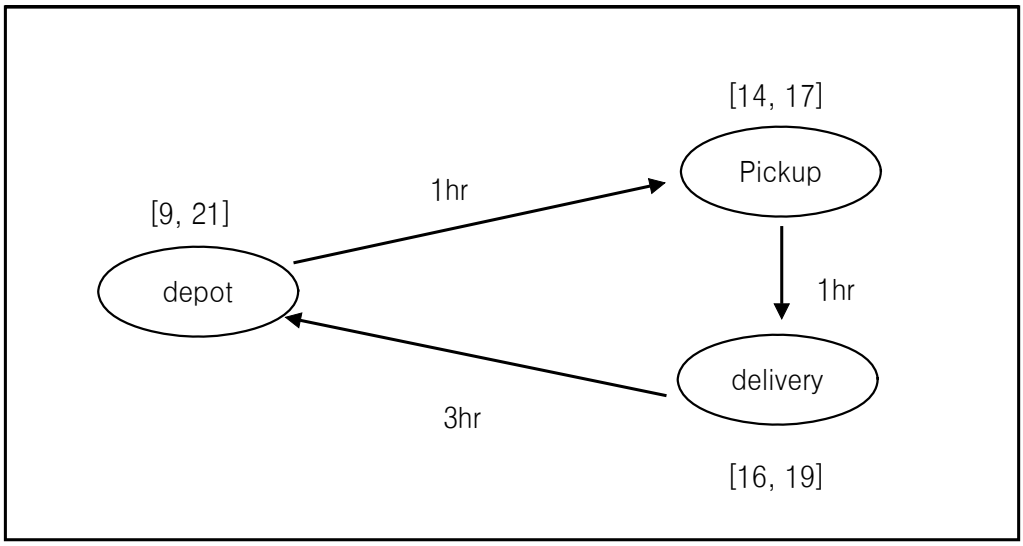

Figure 1. Example

cost per unit time is higher than the regular working cost per unit time.

The following example shows the most important difference between the existing problem and our problem. For the simplicity, only one transportation request is considered and the <Figure 1> displays time windows for locations and travel times between locations.

There are three locations, a depot for a vehicle, a pickup location and a delivery location of a transportation request. Due to the precedence constraint, there is only one feasible route. The vehicle can be used from 9 a.m. till 9 p.m. and the travel time from the depot to the pickup location is one hour. Similarly, time windows of service locations and travel times are interpreted from the figure. Service time is assumed to be zero since it can be added to the travel time. Consider two solutions of which schedules are displayed by <Table 2 ~ Table 3>. Regardless of different schedules, the two solutions have the same objective value for the PDPTW considered by Savelsbergh and Sol (1998), because they use only one vehicle and the travel distances are same. However, they have different working hours. Solution 2 requires only five working hours, whereas solution 1 needs ten working hours including waiting time. That is, solution 2 has smaller objective value than solution 1 in our problem. Solution 1 wastes too much time and it can not be a good solution in the real world even though it is optimal to the existing PDPTW. In addition, if the duration of a route is restricted up to nine hours, solution 1 is not even feasible. The optimization algorithms by Dumas *et al*. (1991) and Savelsbergh and Sol (1998) only provide solution 1 since the algorithms fix the departure time of every vehicle to the earliest arrival time of the corresponding depot. And it is not easy to transform a solution provided by the existing optimization algorithms into a solution optimal to our problem. Therefore, we propose another optimization algorithm for the new PDPTW.

Table 2. Solution 1

	Depot	Pickup Location	Delivery Location	Depot
Arrival Time		10 a.m.	3 p.m.	7 p.m.
Departure Time	9 a.m.	2 p.m.	4 p.m.	

Table 3. Solution 2

	Depot	Pickup Location	Delivery Location	Depot
Arrival Time		3 p.m.	4 p.m.	7 p.m.
Departure Time	2 p.m.	3 p.m.	4 p.m.	

3. Model

We consider different vehicle types and assume that the available vehicles are restricted for each vehicle type. And different travel times are considered according to the vehicle types. It is allowed that a vehicle arrives at a location before the earliest arrival time. Then the vehicle should wait until the earliest arrival time.

But, arrival after the latest arrival time is not allowed. We assume that the service time of a location is zero because it can be easily included into the travel time.

For the branch and price approach, the PDPTW should be divided into the master problem and the subproblem. If we can find all possible routes, the problem can be converted into a problem to decide whether we use the routes or not. The converted problem is called the master problem. The following notation is used to model the master problem:

N the set of all transportation requests

M the set of all vehicle types

m_k the number of available vehicles for vehicle type $k \in M$

Ω_k the set of all feasible routes for vehicle type $k \in M$

δ_{lr}^k = $\begin{cases} 1 & \text{if transportation request } l \text{ is served on route } r \in \Omega_k \\ 0 & \text{o.w.} \end{cases}$

c_r^k = the cost of route $r \in \Omega_k$

The decision variables are as follows:

x_r^k = $\begin{cases} 1 & \text{if route } r \in \Omega_k \text{ is used} \\ 0 & \text{o.w.} \end{cases}$

The following formulation is for the master problem.

$$\text{Min} \sum_{k \in M} \sum_{r \in \Omega_k} c_r^k \; x_r^k \tag{1}$$

$$\text{s.t.} \sum_{k \in M} \sum_{r \in \Omega_k} \delta_{lr}^k \; x_r^k = 1 \quad \text{for all } l \in N \tag{2}$$

$$\sum_{r \in \Omega_k} x_r^k \leq m_k \quad \text{for all } k \in M \tag{3}$$

$$x_r^k \in \{0,\ 1\} \quad \text{for all } k \in M,\ r \in \Omega_k \tag{4}$$

The objective function is represented by (1). Constraints (2) impose that each transportation request must be satisfied exactly once. Constraints (3) represent availability constraints. Feasible route r for vehicle type k corresponds to column vector $(\delta_{1r}^k, \delta_{2r}^k, \cdots, \delta_{|N|r}^k)'$. Since there are generally exponential numbers of columns, it is impractical to enumerate all possible columns. However, we can solve the master problem without enumerating all feasible columns by the branch-and-price algorithm. The master problem with a subset of Ω_k is called by a restricted master problem. The subproblem is a problem to construct a feasible column with the minimum reduced cost. If the reduced cost is negative, the column can be added to the restricted master problem. Otherwise, an optimal solution of the LP relaxation of the restricted master problem is optimal to the LP relaxation of the master problem. The subproblems can be split up according to the vehicle type. We can construct a graph for each subproblem where each location is a node and each path between two locations is an arc. If we duplicate a depot as a starting depot and a ending depot, the subproblem can be regarded as a constrained shortest path problem with time windows. The following notation is used to model the subproblem for vehicle type $k \in M$:

p_l the pickup location of transportation request $l (\in N)$

d_l the delivery location of transportation request $l (\in N)$

$\overline{n_l}$ the number of items for transportation request $l (\in N)$

$\overline{\omega_l}$ the weight of items for transportation request $l (\in N)$

$\overline{v_l}$ the volume of items for transportation request $l (\in N)$

ζ_l the number of extra workers necessary for transportation request $l (\in N)$

k^+ the starting depot of vehicle type $k \in N$

k^- the ending depot of vehicle type $k \in N$

α_k the upper limit of the number of items which can be loaded on vehicle type $k \in N$

β_k the upper limit of the weight of items which can be loaded on vehicle type $k \in N$

χ_k the upper limit of the volume of items which can be loaded on vehicle type $k \in N$

Γ_k the fixed cost for vehicle type $k \in N$

η_k the cost per unit distance of vehicle type $k \in N$

τ_k the driver's regular cost per unit time for vehicle type $k \in N$

λ_k the extra worker's regular cost per unit time for vehicle type $k \in N$

V^k $= (\cup_{l \in N} \{p_l,\ d_l\}) \cup \{k^+,\ k^-\}$

$\overline{u_i}$ the upper limit of the weight for vehicles which can serve at location $i (\in \cup_{l \in N} \{p_l,\ d_l\})$

T the ending time of regular work

γ the overtime weight

a_i the earliest arrival time for service location $i\ (\in\ V^k)$

b_i the latest arrival time for service location $i\ (\in\ V^k)$

d_{ij} the shortest travel distance from node $i\ (\in\ V^k)$ to node $j\ (\in\ V^k)$

t_{ij}^k the shortest travel time from node $i\ (\in\ V^k)$ to node $j\ (\in\ V^k)$ by vehicle type $k \in N$

U_t the upper limit for the schedule duration

U_d the upper limit for the travel distance

B the big number

φ_l the dual variable corresponding to constraints (2) for transportation request $l(\in N)$

ϕ_k the dual variable corresponding to constraints (3) for vehicle type $k \in M$

The decision variables are as follows:

$z_l = \begin{cases} 1 & \text{if transportation request } l(\in N) \text{ is satisfied} \\ 0 & \text{o.w.} \end{cases}$

$x_{ij} = \begin{cases} 1 & \text{if the vehicle travels from node } i(\in V^k) \text{ to node } j(\in V^k) \\ 0 & \text{o.w.} \end{cases}$

t_i the time which the vehicle departs from node $i(\in V^k)$

n_i the number of items on the vehicle when it arrives at node $i(\in V^k)$

w_i the weight of items on the vehicle when it arrives at node $i(\in V^k)$

v_i the volume of items on the vehicle when it arrives at node $i(\in V^k)$

e the necessary extra workers on the vehicle

The cost of a route for vehicle type $k(\in M)$ is divided into three parts as follows:

cost = vehicle cost + distance cost + labor cost
= vehicle cost + distance cost + (driver's cost + extra worker's cost),

where the labor cost of each worker is the sum of regular working cost and overtime working cost. Each part of the cost is calculated as follows:

vehicle cost	Γ_k
distance cost	$\eta_k \sum_{(i,j)\in V^k \times V^k} d_{ij}\, x_{ij}$
driver's regular working cost	$\min\{T - t_{k^+},\, t_{k^-} - t_{k^+}\}\tau_k$
driver's overtime working cost	$\gamma \max\{0,\, t_{k^-} - T\}\tau_k$
extra worker's regular working cost	$\min\{T - t_{k^+},\, t_{k^-} - t_{k^+}\}\lambda_k e$
extra worker's overtime working cost	$\gamma \max\{0,\, t_{k^-} - T\}\lambda_k e$

And the total cost for the route is

$$c = \Gamma_k + \eta_k \sum_{(i,j)\in V^k \times V^k} d_{ij} x_{ij} + (\min\{T - t_{k^+},\, t_{k^-} - t_{k^+}\} + \gamma \max\{0,\, t_{k^-} - T\})(\tau_k + \lambda_k e) \quad (5)$$

And the subproblem for vehicle type $k\,(\in M)$ is as follows:

$$\text{Min} \quad c - \sum_{l\in N} z_l\, \varphi_l - \phi_k \quad (6)$$

$$\text{s.t.} \quad \sum_{i \in V^k \setminus \{k^+\}} x_{k^+ i} = \sum_{i \in V^k \setminus \{k^-\}} x_{i k^-} = 1 \quad (7)$$

$$\sum_{j \in V^k} x_{ij} = \sum_{j \in V^k} x_{ji} = z_l \quad \text{for all } l \in N,\ i \in p_l \cup d_l \quad (8)$$

$$t_{p_l} \le t_{d_l} + B(1 - z_l) \quad \text{for all } l \in N \quad (9)$$

$$t_i + t_{ij}^k \le t_j + B(1 - x_{ij}) \quad \text{for all } i \in V^k, j \in V^k \quad (10)$$

$$n_{k^+} = 0,\ w_{k^+} = 0,\ v_{k^+} = 0 \quad (11)$$

$$n_i \le \alpha_k \quad \text{for all } i \in V^k \quad (12)$$

$$w_i \le \beta_k \quad \text{for all } i \in V^k \quad (13)$$

$$v_i \le \chi_k \quad \text{for all } i \in V^k \quad (14)$$

$$n_{p_l} + \overline{n}_l \le n_j + B(1 - x_{p_l j}) \quad \text{for all } l \in N, j \in V^k \quad (15)$$

$$n_{d_l} - \overline{n}_l \le n_j + B(1 - x_{d_l j}) \quad \text{for all } l \in N, j \in V^k \quad (16)$$

$$w_{p_l} + \overline{w}_l \le w_j + B(1 - x_{p_l j}) \quad \text{for all } l \in N, j \in V^k \quad (17)$$

$$w_{d_l} - \overline{w}_l \le w_j + B(1 - x_{d_l j}) \quad \text{for all } l \in N, j \in V^k \quad (18)$$

$$v_{p_l} + \overline{v}_l \le v_j + B(1 - x_{p_l j}) \quad \text{for all } l \in N, j \in V^k \quad (19)$$

$$v_{d_l} - \overline{v}_l \le v_j + B(1 - x_{d_l j}) \quad \text{for all } l \in N, j \in V^k \quad (20)$$

$$\beta_k z_l \le \overline{u}_{p_l} \quad \text{for all } l \in N \quad (21)$$

$$\beta_k z_l \le \overline{u}_{d_l} \quad \text{for all } l \in N \quad (22)$$

$$a_i \le t_i \le b_i \quad \text{for all } i \in V^k \quad (23)$$

$$t_{k^-} - t_{k^+} \le U_t \quad (24)$$

$$\sum_{(i,j) \in V^k \times V^k} d_{ij} x_{ij} \le U_d \quad (25)$$

$$e \ge \zeta_l z_l \quad \text{for all } l \in N \quad (26)$$

$$x_{ij} \in \{0, 1\} \quad \text{for all } i \in V^k, j \in V^k \quad (27)$$

$$z_l \in \{0, 1\} \quad \text{for all } l \in N \quad (28)$$

$$t_i \geq 0 \quad \text{for all } i \in V^k \tag{29}$$
$$e \geq 0, \text{ integer} \tag{30}$$
$$n_i \geq 0, \text{ integers} \quad \text{for all } i \in V^k \tag{31}$$
$$w_i \geq 0 \quad \text{for all } i \in V^k \tag{32}$$
$$v_i \geq 0 \quad \text{for all } i \in V^k \tag{33}$$

Constraints (8) and (9) are pairing and precedence constraints. Vehicle capacity constraints are expressed by constraints (12), (13), and (14). Location capacity constraints are presented by constraints (21) and (22). Constraints (23) are time windows constraints. Constraints (24) and (25) are corresponding to the duration constraint and the distance constraint respectively. Constraints (10) mean time compatibility constraints and constraints (15) ~ (20) are capacity compatibility constraints. Compatibility constraints prevent cycle in a route.

4. Algorithm

In this paper, a branch-and-price algorithm is presented for the PDPTW. We optimize the linear programming (LP) relaxation of the restricted master problem instead of the LP relaxation of the master problem because there are too many feasible columns to enumerate. Although it is not trivial to construct an initial subset of columns for the restricted master problem, we can initialize it by the two-phase method. If any column with negative reduced cost is found, we can add it to the restricted master problem and also reoptimize the LP relaxation of the problem. Otherwise, a current optimal solution to the LP relaxation of the restricted master problem is an optimal solution to the LP relaxation of the master problem because all possible columns have nonnegative reduced cost. Therefore, we repeat the procedure to find any columns of the negative reduced cost until no more columns with negative reduced costs are found. If an optimal solution of the LP relaxation of the master problem is not integral, we need to explore a branch-and-bound tree. We have to generate columns at each branch-and-bound node, too.

The enumeration method was extended from the dynamic programming algorithm which was proposed by Dumas *et al*. (1991) to solve the subproblem. Preprocessing steps such as the shrinking of the time windows and the elimination of the inadmissible arcs are performed before the enumeration process starts(For details, see Dumas *et al*., 1991). Location capacity constraints can be ensured by eliminating the inadmissible arcs. The following notation is used:

P_i^q	path q from the starting depot to node i
S_i^q	the set of nodes visited on path q
$R(S_i^q)$	the set of nodes which have to be visited on path q between node i and the ending depot
T_i^q	the departure time from node i on path q
X_i^q	the departure time from the starting depot on path q
I_i^q	$\begin{cases} 0 & \text{if the departure time from the starting depot on path } q \text{ can be deferred,} \\ 1 & \text{o.w.} \end{cases}$
Z_i^q	the travel distance from the starting depot to node i on path q
a_i^k	the earliest arrival time at node i by vehicle type $k \in M$ after shrinking time windows
b_i^k	the latest arrival time at node i by vehicle type $k \in M$ after shrinking time windows

Each path P_i^q corresponds to label $(S_i^q,\ R(S_i^q),\ T_i^q,\ X_i^q,\ I_i^q,\ Z_i^q)$. A label contains all information to verify satisfaction of constraints. And the enumeration method for generating columns is as follows:

4.1 Enumeration Method

Step 1. Initialize a path list (L), the minimum reduced cost (*mrc*), and the minimum reduced cost path (*mrcp*) as follows:
$L = \{P_{k^+}^0\}$, where the label of path $P_{k^+}^0$ is $(\{k^+\}, \varnothing, a_{k^+}, a_{k^+}, 0, 0)$.
mrc : = 0.
mrcp : = $\varnothing$.

Step 2. If $L = \varnothing$, then stop.
Otherwise, choose a path from L.

Step 3. Given a path P_i^q with label $(S_i^q,\ R(S_i^q),\ T_i^q,\ X_i^q,\ I_i^q,\ Z_i^q)$,
Case 3.1 $i \neq k^-$
→ Go to step 4.
Case 3.2 $i = k^-$ and P_i^q violates pairing and duration constraints
→ Discard P_i^q and go to step 2.
Case 3.3 $i = k^-$, P_i^q satisfies pairing and duration constraints, and the reduced cost of P_i^q < *mrc*
→*mrc* : = the reduced cost of P_i^q, *mrcp* : = P_i^q and go to step 2.

Case 3.4 $i = k^-$, P_i^q satisfies pairing and duration constraints, and the reduced cost of $P_i^q \geq mrc$
→ Discard P_i^q and go to step 2.

Step 4. Call procedure *path_extension* (P_i^q, j) for all existing arc (i, j) satisfying $j \notin S_i^q$. Then, discard P_i^q and go to step 2.

The basic idea of the enumeration method is extending paths from the starting depot by adding arcs one by one. During the method, pairing and duration constraints can be verified. At step 3, if node i is k^- and set $R(S_i^q)$ is not empty**,** it means that path P_i^q violates the pairing constraints. And if node i is k^- and $T_i^q - X_i^q > U_t$, then path P_i^q violates the duration constraint. If path P_i^q is found to be infeasible, it is discarded and other path is picked out from paths list *L*. If path P_i^q is a feasible completed path, *mrc* and *mrcp* are updated. Procedure *path_extension* (P_i^q, j) at step 4 is for obtaining a new extended path. If arc (i, j) can be added to path P_i^q, the procedure provides extended path $P_j^{q'}$ with label $(S_j^{q'}, R(S_j^{q'}), T_j^{q'}, X_j^{q'}, I_j^{q'}, Z_j^{q'})$.

Procedure *path_extension*(P_i^q, j)

1: Create path $P_j^{q'}$ with label. ($\varnothing$, $\varnothing$, *null*, *null*, *null*, *null*)
2: $S_j^{q'} := S_i^q \cup \{j\}$.
3: If path $P_j^{q'}$ violates the vehicle capacity constraint, discard path $P_j^{q'}$ and stop.
4: If $Z_i^q + d_{ij} \leq U_d$, $Z_j^{q'} := Z_i^q + d_{ij}$.
Otherwise, discard path $P_j^{q'}$ and stop.
5: If $l \in N$ and $j \in p_l$, $R(S_j^{q'}) := R(S_j^q) \cup \{d_l\}$.
If $j \in R(S_i^q)$, $R(S_j^{q'}) := R(S_i^q) \setminus \{j\}$.
Otherwise, discard path $P_j^{q'}$ and stop.
6: Case 6.1 $I_i^q = 0$, $T_i^q + t_{ij}^k < a_j^k$
→ $T_j^{q'} := a_j^k$ and call procedure *determent*$(P_j^{q'})$.
Case 6.2 $I_i^q = 1$, $T_i^q + t_{ij}^k < a_j^k$
→ $T_j^{q'} := a_j^k$, $X_j^{q'} := X_i^{q'}$, $I_j^{q'} := I_i^q$
Case 6.3 $a_j^k \leq T_i^q + t_{ij}^k \leq b_j^k$
→ $T_j^{q'} := T_i^q + t_{ij}^k$, $X_j^{q'} := X_i^q$, $I_j^{q'} := I_i^q$
Case 6.4 $T_i^q + t_{ij}^k > b_j^k$
→ Discard path $P_j^{q'}$ and stop.
7: L := L + $\{P_j^{q'}\}$

This procedure verifies vehicle capacity, distance, precedence and time window constraints and provides new label for path $P_j^{q'}$. The vehicle capacity constraint can be easily verified by using set $S_j^{q'}$. And the distance constraint is verified at line 4. The precedence constraints are verified by set $R(S_j^{q'})$. If node j is a pickup node of a transportation request, the delivery node of the transportation request is added to the set $R(S_j^{q'})$. If node j is a delivery location which is an element of set $R(S_i^q)$, it is removed from the set. If node j is a delivery location which does not belong to set $R(S_i^q)$, path $P_j^{q'}$ violates the precedence constraint. If path $P_j^{q'}$ is found to be infeasible, the above procedure should be stopped. Line 6 is for determining time components and verifying the time window constraints. If the starting time of the path is fixed, components $T_j^{q'}$, $X_j^{q'}$ and $I_j^{q'}$ are determined directly. And also, the time window constraints can be verified. But, if the vehicle arrives at node j before the earliest arrival time and the starting time of the path can be postponed, $X_j^{q'}$ is reckoned backward from $T_j^{q'}$. Procedure *determent* $(P_j^{q'})$ is used for updating $X_j^{q'}$ and $I_j^{q'}$.

Procedure *determent* $(P_j^{q'})$

1: $I_j^{q'} := 0$, $t_j = T_j^{q'}$.
2: m := node *j*
3: while $(m \neq k^+)$ {
4: *n* := node visited just before node m on path $P_j^{q'}$
5: $t_n := \begin{cases} t_m - t_{nm}^k & \text{if } a_n^k \leq t_m - t_{nm}^k \leq b_n^k \\ b_n^k & \text{if } t_m - t_{nm}^k > b_n^k \end{cases}$
6: $I_j^{q'} := 1$ if $t_n = b_n^k$.
7: *m* := node *n*.}

Due to this recalculating procedure, we do not need to fix the starting time of a path and we can consider the objective function and constraints relating to the schedule duration. If path $P_{k^-}^q$ is finally found from the enumeration method, the number of extra workers can be determined from the path information.

If an optimal solution of the LP relaxation of the master problem is fractional, we solve the restricted master problem using CPLEX callable mixed integer library. The integral solution can provide an upper bound. And then, we explore the branch-and-bound tree. Supposing that *x* is fractional, and $y_{ij} = \sum_{k \in M} \sum_{r \in \Omega_k}$

$\delta_{ir}^k\ \delta_{jr}^k\ x_r^k$, there must be two requests $i,\ j \in N$ satisfying $0 < y_{ij} < 1$. Then we can divide feasible region into two subsets characterized by $y_{ij} = 0$ and $y_{ij} = 1$ (Savelsbergh and Sol 1998; Barnhart *et al*. 1998). We can generate columns at any branch-and-bound node if we use an adjusted enumeration method which is very similar to the previous one.

5. Computational Experiments

The branch-and-price algorithm was coded in C and CPLEX 8.1 callable library was used to solve LP. The algorithm was tested on a Pentium PC (2.4GHz). We randomly generated forty instances of the PDPTW. The instances are categorized into four problem sets according to the number of service locations and ten instances were generated for each problem set. The problem sets are A20, A30, A40 and A50. Each number denotes the number of service locations. For example, an instance of problem set A20 includes twenty service locations, i.e. it considers ten transportation requests. It is not easy to make a feasible instance since the PDPTW has many constraints. Therefore, we started to generate instances by making a pool of one hundred and three feasible transportation requests. Service locations and depots were selected within a square of size 100 by 100. For each transportation request, two numbers were chosen for the earliest arrival times. The smaller one was assigned for the pickup location and the other was given to the delivery location. The time windows for service locations were randomly chosen within a time interval between fifteen and sixty five. After making the pool of transportation requests, we could randomly select transportation requests among the pool. The travel distances and the travel times between locations were determined from the Euclidean distance. The planning period was from zero to three hundred. Working until one hundred fifty unit time was considered as the regular working and working after one hundred fifty unit time was considered as the overtime working. The overtime weight was randomly selected from one point two to two. The travel distance and the schedule duration were bounded by two hundred. Therefore, if a vehicle departs from the depot, it should be back to the depot in two hundred unit time. The vehicle types were classified by the capacity and the depot. The depots were randomly chosen in the same way to service locations. The capacities were decided from seventy to one hundred sixty. For each instance, two capacities and two depots were decided and by the combination, four vehicle types were considered. The number of available vehicles for each vehicle type was the ceiling of the half of the number of transportation requests. And the location capacity of each service location is the sum of the minimum vehicle capacity of the instance and some randomly generated number. Unfortunately, this

Table 4. The test result of problem set A20

	Z_{IP}	Z_{LP}	GAP(%)	#B&B	#COLS	TIME(sec)
A20.1	5075.46	5075.46	0.00	1	32	0.75
A20.2	4455.56	4455.56	0.00	1	36	0.92
A20.3	9182.64	9182.64	0.00	1	16	0.22
A20.4	5238.72	5238.72	0.00	1	33	0.63
A20.5	8579.42	8448.83	1.52	11	217	1.45
A20.6	3117.53	3117.53	0.00	1	26	2.39
A20.7	6523.94	6523.94	0.00	1	38	2.78
A20.8	4848.82	4848.82	0.00	1	32	1.32
A20.9	4771.48	4771.48	0.00	1	34	3.86
A20.10	5670.30	5670.30	0.00	1	27	1.33
avg.	5746.39	5733.33	0.15	2.00	49.10	1.57

Table 5. The test result of problem set A30

	Z_{IP}	Z_{LP}	GAP(%)	#B&B	#COLS	TIME(sec)
A30.1	6575.80	6534.58	0.63	7	224	17.59
A30.2	9596.72	9596.72	0.00	1	50	5.25
A30.3	5881.52	5863.76	0.30	9	214	30.72
A30.4	5920.60	5764.23	2.64	27	808	211.51
A30.5	4513.54	4513.54	0.00	1	59	122.16
A30.6	5052.11	5022.51	0.59	5	117	60.95
A30.7	7908.91	7509.48	5.05	49	1682	273.38
A30.8	8272.55	8217.62	0.66	9	267	140.70
A30.9	10350.25	9638.87	6.87	605	24039	1735.27
A30.10	5330.65	5096.00	4.40	75	3516	1157.59
avg.	6940.27	6775.73	2.11	78.80	3097.60	375.51

Table 6. The test result of problem set A40

	Z_{IP}	Z_{LP}	GAP(%)	#B&B	#COLS	TIME(sec)
A40.1	13080.87	13080.87	0.00	1	58	15.02
A40.2	17451.58	17451.58	0.00	1	45	4.45
A40.3	11284.77	11094.88	1.68	39	2408	595.86
A40.4	11380.05	11223.15	1.38	41	3028	5269.67
A40.5	9925.79	9702.66	2.25	43	2366	1769.39
A40.6	10396.09	10321.12	0.72	9	478	725.84
A40.7	8842.94	9713.54	1.46	21	982	95.73
A40.8	15081.55	14254.53	5.48	305	17698	2013.59
A40.9	16001.94	15940.86	0.38	11	642	140.42
A40.10	9729.63	9490.87	2.45	75	3752	449.41
avg.	12317.52	12227.41	1.58	54.60	3145.70	1107.94

Table 7. The test result of problem set A50

	Z_{IP}	Z_{LP}	GAP(%)	#B&B	#COLS	TIME(sec)
A50.1	53782.25	52165.83	3.01	13	2196	6037.17
A50.2	44439.35	44439.35	0.00	1	93	90.41
A50.3	66762.67	66762.67	0.00	1	140	140.94
A50.4	73560.79	73560.79	0.00	1	150	153.55
A50.5	66952.65	65141.55	2.71	3	357	1674.97
A50.6	49866.84	49866.84	0.00	1	139	246.72
A50.7	52762.00	52762.00	0.00	1	233	3125.95
A50.8	45342.09	45342.09	0.00	1	131	248.89
A50.9	41889.78	40462.66	3.41	3	1248	2273.31
A50.10	67836.55	67836.55	0.00	1	233	3611.16
avg.	56319.50	55834.03	0.91	2.60	492.00	1760.31

procedure does not guarantee the feasibility of the generated instances due to the complicated constraints. Therefore, we got computational results of feasible instances by lots of trials. <Table 4 ~ Table 7> show us the computational results.

6. Conclusions

Even though many heuristic approaches for the PDPTW consider the minimization of the total waiting time or the schedule duration, neither of the two existing optimization algorithms can handle those objectives since the algorithms fix the starting time of a route to the earliest starting time of the vehicle which serves the route. In this paper, we considered the PDPTW of which objective was the minimization of the number of used vehicles, the total travel distance, and the total labor costs based on the schedule duration. And some realistic constraints were added to the problem, too. The branch and price algorithm was extended from the existing optimization algorithms and the enumeration method for the subproblem adopted the reverse calculating procedure to update the starting time of each route. We tested the algorithm on randomly generated instances and the result showed that the algorithm can provide optimal solutions of instances of moderate size in proper times.

This work can be extended in two ways. The first one is relating to the GPDP. As we mentioned in section 1, Savelsbergh and Sol (1998)'s work does not cover the exact GPDP. Even Kang (2004) presented an algorithm for the GPDP, the problem only considered transportation specifying one pickup location and one or more delivery locations without predetermined order. Therefore, an algorithm for the GPDP can be studied. The second is improvement of the performance of the algorithm. Even though this problem is very complicated and hard, the industrial applications request good solutions in a short time. Therefore, some techniques such as primal heuristic approaches are necessary.

References

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P. and Vance, P. H. (1998), Branch-and-Price : Column Generation for Solving Huge Integer Programs, *Operations Research*, **46**, 316-329.

Ball, M. O., Magnanti, T. L., Monma, C. L. and Nemhauser, G. L. (1995), *Network Routing, Handbooks in Operations Research and Management Science*, **8**, 35-139, Elsevier Science B. V., Amsterdam, Netherlands.

Dumas, Y., Desrosiers, J. and Soumis, F. (1991), The Pickup and Delivery Problem with Time Windows, *European Journal of Operational Research*, **54**, 7-22.

Jih, W. R, Kao, C. Y. and Hsu, J. Y. (2002), Using Family Competition Genetic Algorithm in Pickup and Delivery Problem with Time Window Constraints, *Proceedings of the 2002 IEEE International Symposium on Intelligent Control*.

Kammarti, P., Hammadi, S., Borne, P. and Ksouri, M. (2004), A New Hybrid Evolutionary Approach for the Pickup and Delivery Problem with Time Windows, *2004 IEEE International Conference on Systems, Man and Cybernetics*.

Kang, J. (2004), An Optimization Algorithm for a Generalized Pickup and Delivery Problem with Time Windows, *Korea Advanced Institute of Science and Technology*, Thesis.

Lau, H. C. and Liang (2001), Z. Pickup and Delivery with Time Windows : Algorithms and Test Case Generation, *13th IEEE International Conference on Tools with Artificial Intelligence*.

Li, H. and Lim. A. (2001), A metaheuristic for the pickup and delivery problem with time windows, *13th IEEE International Conference on Tools with Artificial Intelligence*.

Nanry, W. P. and Barnes, J. W. (2000), Solving the Pickup and Delivery Problem with Time Windows Using Reactive Tabu Search, *Transportation Research*, Part B, **34**, 107-121.

Savelsbergh, M. W. P. and Sol, M. (1995), The General Pickup and Delivery Problem, *Transportation Science*, **29**, 17-29.

Savelsbergh, M. W. P. and Sol, M. (1998), DRIVE: Dynamic Routing of Independence Vehicles, *Operations Research*, **46**(4), 474-490.