Journal of the Korean Institute of Industrial Engineers
Vol. 33, No. 3, pp. 303-311, September 2007.

A Heuristic Algorithm for Multi-path Orienteering Problem with Capacity Constraint

Hark Hwang[1†] · Keum Ae Park[2] · Yong Hui Oh[3]

[1]Department of Industrial Engineering, KAIST, Daejeon 305-701
[2]Samsung Card / [3]Department of Industrial System Engineering, Daejin University

용량제약이 있는 다경로 오리엔티어링 문제의 해법에 관한 연구

황 학[1] · 박금애[2] · 오용희[3]

[1]한국과학기술원 산업공학과 / [2]삼성카드 / [3]대진대학교 산업시스템공학과

This study deals with a type of vehicle routing problem faced by manager of some department stores during peak sales periods. The problem is to find a set of traveling paths of vehicles that leave a department store and arrive at a destination specified for each vehicle after visiting customers without violating time and capacity constraints. The mathematical model is formulated with the objective of maximizing the sum of the rewards collected by each vehicle. Since the problem is known to be NP-hard, a heuristic algorithm is developed to find the solution. The performance of the algorithm is compared with the optimum solutions obtained from CPLEX for small size problems and a priority-based Genetic Algorithm for large size problems.

Keywords: Heuristic, Multi-path Orienteering Problem, Time and Capacity Constraints

1. Introduction

In holiday seasons such as Thanksgiving and year-end sales period with Christmas, managers of some department stores face a tremendously large amount of delivery requests of customers. The customers in general want to have their purchased products delivered to designated location by a given date. One doable way for the managers is to get delivery support from some of their employees who commute by car. They are asked to deliver customers' orders on their way home from the work place. Now, the manager's problem is how to assign the delivery requests to each employee. <Figure 1> shows an example of graphic solution for the problem with four helpful employees, A, B, C and D. White dots in the figure denote the locations of customers while black dots the locations of employees' houses. For instance, employee A is required to arrive at his house no later than three hours after his departure at the depot. His car is known to have the loading capacity of 2.0m^3. A sequence of white dots connected by arrow signs forms a path and only the customers on each path are visited by the corresponding employee. We find that the problem is similar to the orienteering problem (OP), the team orienteering problem (TOP), the maximum

This work was supported by Jungseok Logistics Foundation Grant.
† Corresponding author : Professor Hark Hwang, Department of Industrial Engineering, KAIST, 373-1 Kuseong-dong, Yuseong-gu, Daejeon 305-701, Korea, Fax : +82-42-869-3110, E-mail : harkhwang@kaist.ac.kr
Received May 2007; revision received July 2007; accepted July 2007.

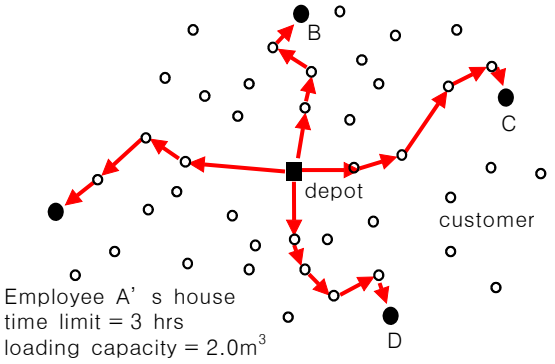

Figure 1. An example of graphic solution of MPOPCC

collection problem (MCP), and the multiple-tour maximum collection problem (MTMCP). Thus we name the manager's problem 'the multi-path orienteering problem with capacity constraint (MPOPCC). MPOPCC and the problems mentioned above are similar in the following aspects: (1) the concept of "reward" exists and (2) not all the customers need to be visited. On the other hand, they differ in the sense that the destination, time limit, and capacity of each vehicle are not necessarily identical in MPOPCC, while they are the same in the other problems.

Note that MPOPCC is also different from the vehicle routing problem (VRP). In ordinary VRP, all customers need to be visited with the objective of finding a set of vehicle routes in a way to minimize the total cost without exceeding vehicle capacities. Thus, existing algorithms developed for VRP cannot be directly applicable to the MPOPCC. The orienteering problem (OP) was studied by many researchers. The objective of OP is to maximize the total collected reward within a prescribed time limit. Laporte and Martello (1990) developed an exact solution method using a branch and bound method which can solve up to 90 vertices in tested problems within 100 seconds. Leifer and Rosenwein (1994) proposed a procedure to obtain upper bounds by solving three successive linear programs tightening the LP relaxation by adding constraints and valid inequalities. Fischetti, Gonzalez, and Toth (1998) developed a branch-and-cut algorithm for finding an optimal OP solution. They proposed an effective way to use a family of cuts within the overall branch-and-cut framework. It is known that OP is NP-hard (Golden, Levy and Vohra, 1987) and so many different heuristic solutions appeared in the literature. Tsiligirides (1984) developed two heuristics, a stochastic algorithm using a Monte Carlo technique and a deterministic algorithm modifying Wren and Holiday's (1972) method for a vehicle-scheduling problem. Hayes and Norman (1984) used a simple functional equation of dynamic programming. Golden, Levy, and Vohra (11) developed a heuristic procedure using a 'bang for buck' insertion and a center of gravity improvement. With the center of gravity idea and Tsiligirides's randomization concept, Golden, Wang, and Liu (1988) proposed a multifaceted heuristic. Keller (1989) modified his algorithm for the multi-objective vending problem (Keller and Goodchild, 1988) to solve the orienteering problem. The algorithm consists of a path-construction stage using a desirability measure and an improvement stage. Chao, Golden and Wasil (1996) presented a heuristic that consists of two-point exchange, one-point movement, clean up, and reinitialization. Tasgetiren and Smith (2000) presented a genetic algorithm to solve the OP. Liang, Kulturel-Konak and Smith (2002) compared an ant colony optimization method and tabu search for the OP. Mocholi, Jaen and Canos (2005) proposed a distributed ant colony algorithm to solve large scale OP which approach is based on the ideas of Grid Computing so that large instances of OP can be solved collaboratively.

The maximum collection problem (MCP) is a special case of OP where the start and end point are the same. MCP is a routing problem where the objective is to maximize the sum of the rewards collected at the customers visited. Kataoka and Morito (1988) proposed a branch and bound procedure to solve optimally. Ramesh, Yoon and Karwan (1992) used Lagrangian relaxation along with improvement procedures within a branch and bound method to solve large, randomly generated test problems that contain as many as 150 nodes. Deitch and Ladany (2000) utilized and modified Tsiligirides's heuristic to solve the one-period bus touring problem which consists of determining the optimal subset of tourist sites to be visited and scenic routes to be traversed between a start and end point that both coincide.

There are other problems considering multiple paths or tours that have received relatively little attention than the single path or tour. The team orienteering problem (TOP) presented by Chao, Golden and Wasil (1996) is an extended version of OP. In

TOP, the start and end points are different and there is more than one vehicle. The objective of TOP is to maximize the total team rewards without violating a specified time limit of each vehicle. Their heuristic is based on the notion of record-to-record improvement similar with their heuristic algorithm for the OP except reinitialization step. They compared their results with modified Tsiligirides's stochastic algorithm for OP.

There is another type of multi-path problem, the multiple-tour maximum collection problem (MTMCP). The objective of MTMCP is to maximize the total reward collected on all of the tours without exceeding the time constraints where the start and end nodes are the same. Butt and Cavalier (1994) developed a heuristic solution procedure. They found the problem from the athletic department of a college. The campus; they had to visit during the specified hours of a day and limited number of days. Butt and Ryan (1999) presented an optimal solution procedure for MTMCP. This procedure is based on a generalized set-partitioning formulation and uses constraint branching and tour storage techniques to improve solution time. Their procedure works well when the number of nodes visited in any tour is relatively small. Recently, Tang and Miller-Hooks (2005) used one of the meta-heuristic, tabu search embedded in an adaptive memory procedure.

Recently, Hwang, Park and Gen (2006) proposed a priority-based genetic algorithm (pGA) to solve a type of MPOPCC that does not consider capacity constraint. In this study, we develop the mathematical model of MPOPCC and then propose a heuristic solution procedure. The performance of the heuristic is compared with the solutions of various test problems obtained from CPLEX and pGA.

2. Mathematical Formulation

We use an undirected graph $G = \{N, A\}$ to formulate MPOPCC with K number of vehicles into the mathematical model, where $N = \{0, 1, 2, \cdots, n\}$ is a set of nodes and $A \subseteq N \times N$ is a set of arcs. Each node i in N is associated with a reward $r_i \geq 0$. N can be partitioned into three subsets of nodes, i.e., $N = N_S \cup N_C \cup N_D$, where $N_S = \{0\}$, $N_C = \{1, 2, \cdots, n-K\}$ and $N_D = \{n-K+1, n-K=2, \cdots, n\}$. Node 0 in N_S represents the depot (or department store) from which all K vehicles leave to visit customers. N_C is a set of customer nodes associated with delivery request while N_D is a set of destination nodes (i.e., employees' houses) of vehicles. Each arc in A is associated with a symmetric and nonnegative value of t_{ij}, the time required for the vehicle to travel between node i and j. Euclidean distance with constant speed of vehicles is assumed for t_{ij}. We want to find a set of K paths, in which each path starts from node 0 and ends at a given node in N_D in a way to maximize the total rewards collected by the vehicles. In the solution of MPOPCC, not all the customers need to be visited due to the time and capacity constraints. The reward of a node in N_C is awarded only once. In this study, we assume that customer node can be visited by at most one vehicle to facilitate the development of the model. As constraint, the total time required to visit the nodes in each path should not exceed a specified time limit, $T^k + A^k$. As another constraint, the total volume of the items delivered in each path should not exceed a given capacity of each vehicle. Note that for the second constraint only the amount of volume is counted while giving no consideration to the physical configuration of each item to be delivered.

The following notations are adopted to develop the model:

- n the number of nodes excluding the depot node
- K the number of available vehicles at the depot
- r_i reward associated with customer node i. It could be the monetary value of the items to be delivered to node i or any positive real number.
- t_{ij} travel time from node i to node j
- T^k travel time from the depot node to the destination node of vehicle k
- A^k overtime allowed for vehicle k
- V_i volume of the items to be delivered to customer node i
- C^k volume capacity of vehicle k

Also, the following 0-1 decision variables are introduced.

$x_{ij}^k = 1$ if arc(i, j) is in the path of vehicle k

otherwise, 0

$y_i = 1$ if node i is in any path

otherwise, 0

We formulate MPOPCC as follows:

(P) max $$\sum_{i \in N_c} r_i y_i \quad (1)$$

subject to :

$$\sum_{i \in N_S \cup N_C} \sum_{k=1}^{K} x_{ij}^k = y_j \qquad \forall j \in N_C \quad (2)$$

$$\sum_{j \in N_C \cup N_D} \sum_{k=1}^{K} x_{ij}^k = y_j \qquad \forall i \in N_C \quad (3)$$

$$\sum_{i \in N_S \cup N_C} x_{ij}^k = \sum_{l \in N_C \cup N_D}^{K} x_{jl}^k \qquad \forall k, j \in N_C \quad (4)$$

$$\sum_{j \in N} x_{0,j}^k = 1 \qquad \forall k \quad (5)$$

$$\sum_{i \in N_S \cup N_C} x_{i,n-k+1}^k = 1 \qquad \forall k \quad (6)$$

$$\sum_{(i,j) \in A} t_{ij} x_{ij}^k \leq T^k + A^k \qquad \forall k \quad (7)$$

$$\sum_{(i,j) \in A} V_j x_{ij}^k \leq C^k \qquad \forall k \quad (8)$$

$$\sum_{i \in U} \sum_{j=U} x_{ij}^k \leq |U| - 1 \quad U \subset N \setminus \{0\}; |U| \geq 2; \qquad \forall k \quad (9)$$

$$x_{ij}^k, y_i \in \{0, 1\} \qquad \forall (i,j) \in A \quad (10)$$

The objective function (1) maximizes the total rewards collected from visiting modes by K vehicles. Constraints (2) and (3) are the so-called assignment constraints, which imply that for any customer node at most only one vehicle can visit as well as depart from the node. Constraint (4) is related with the flow conservation. It implies that the vehicle should come out of the node if it enters a customer node. It also guarantees the continuity of each path. Constraints (5) and (6) ensure that each vehicle should start from the depot node and end at the destination node, respectively. Constraint (7) is for the time restriction and constraint (8) is for the capacity restriction of the vehicles. Constraint (9) is for preventing sub-tour. If the index k is additionally eliminated from (P), the mathematical formulation of MPOPCC is the same as that of OP, which implies that the complexity of MPOPCC equals at least that of OP. One may try to solve MPOPCC by applying an algorithm developed for OP K times consecutively. But the approach has a shortcoming of ignoring the sequence dependency and may result in poor solution.

3. Heuristic Algorithm

MPOPCC is shown to have a more complicated problem structure than OP which is known to be NP-hard. Therefore, we develop a heuristic algorithm of MPOPCC in this section. It is a construction type algorithm consisting of four steps. First, utilizing the concept of ellipse (Keller, 1989), k number of ellipses is drawn taking the depot node and destination node as two foci of each ellipse and time limit, $T^k + A^k$, as the length of major axis. Only the customer nodes inside a given ellipse become candidates in constructing the corresponding path. Note that any path containing nodes outside of the corresponding ellipse inevitably violates the time limit constraint. Second, K number of paths is first initialized by connecting the depot node to each destination node and then augmented. Third, we select an unrouted customer node and insert it in the current partial paths. Finally, the selection and insertion steps are repeated until the paths are full with respect to the time limit and/or capacity constraint.

3.1 Selection and Insertion of Node

An unrouted customer node is selected based on the value of the priority function which is obtained by combining three different evaluation functions. Once the customer node with the largest priority function value is selected, then its insertion location is determined based on IT_j. Suppose node j is selected and then inserted between two adjacent nodes, nodes i and l, in the current partial path. The additional vehicle travel time occurred by the insertion can be expressed as $IT_{ijl} = t_{ij} + t_{jl} - t_{il}$ $\forall i$ and l. Let IT_j be the minimum among IT_{ijl} and will be called the minimum required insertion time of node j. In our heuristics, the arc associated with IT_j is selected as the insertion location. The selection of unrouted customer node to be added to paths is related with the question of "Does that

particular customer node provide enough rewards considering the additional travel time and vehicle capacity required for the visit?" In this study the factors chosen to be criteria for the selection process are as follow: (1) (relative amount of reward) relative amount of reward that can be collected, (2) (relative length of additional driving time). relative length of additional driving time needed considering the available operation time of vehicle, and (3) (relative amount of capacity required) relative amount of additional capacity needed considering the capacity remained. We develop the evaluation functions corresponding to the factors above and they are

$$fR^k(i) = \frac{\sum_{j \in SC(i,\, r)} r_j}{\sum_{i \in SE^k} r_i}, \quad 0 \le f \le 1 \tag{11}$$

$$fD^k(i) = \frac{IT_i}{time\,remained^k} \tag{12}$$

$$fC^k(i) = \frac{V_i}{capacity\ \ remained^k} \tag{13}$$

Let SE^k be the set of customer nodes contained in ellipse *k*. Also, let $SC(i, r)$ be the set of customer nodes in the circular neighborhood centered at node *i* with radius *r* % (see <Figure 2>). In fR, we consider not only the reward of the node *i* but also those of the neighboring nodes in $SC(i, r)$. The denominator implies the sum of the rewards in the ellipse *k* and thus $fR^k(i)$ measures relative size of rewards associated with node *i* and its neighbor nodes in ellipse *k*. The numerator of fD is the minimum insertion time of node *i* while the denominator indicates the overtime available of employee k at the time of evaluation. The numerator of fC is the volume of the items to be delivered to customer *i* while the denominator is the remaining capacity of the vehicle *k* at the time of evaluation

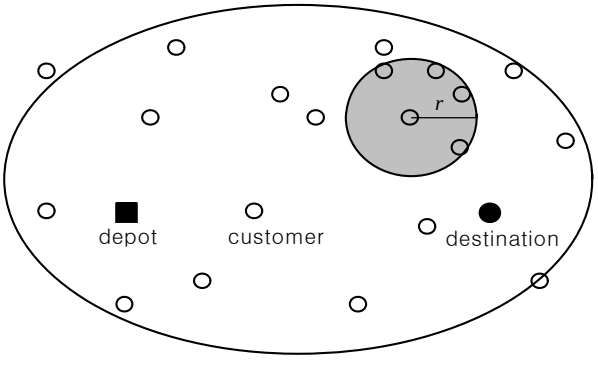

Figure. 2 Graphic representation of an evaluation function associated with reward

3.2 Priority Functions

At each construction step of the paths, we evaluate all the unrouted customers with three evaluation functions of (11), (12) and (13). Now, these function values have to be integrated such that the result provides a basis for the selection of the most qualified node. Among various possible functional forms, this study formulates the following two (will be called 'priority function') and examines the performances.

$$F^1 = \frac{fR^\alpha}{fD^\beta \times fC^r} \tag{14}$$

$$F^2 = fR^\alpha \times (1 - fD)^\beta \times (1 - fC)^r \tag{15}$$

where α, β and γ are positive constants.

Note that the priority function has a larger value when the relative size of reward becomes bigger while either the relative amount of additional driving time or capacity required becomes smaller. For selection procedure, for each unrouted customer we calculate the priority function vector, column vector of size *K*, in which the *i*th element is the priority function value associated with vehicle *i*. Note that the value of the *i*th element becomes zero when the node of unrouted customer is outside of ellipse *i*. The node with the largest priority function value

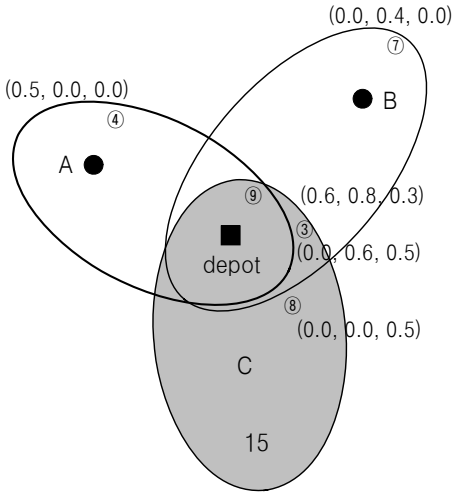

Figure 3. Example of priority vectors

is selected and inserted between the two adjacent nodes associated with IT_j. The procedure is repeated until no more insertion is possible. <Figure 3> illustrates an example of the priority vectors with three employees, A, B, and C. In this case, 0.8 is the largest and so customer 9 is selected and inserted in the arc associated with IT_9 in the current path of vehicle B.

The paths of vehicles for (P) can be constructed using the following procedure.

Priority algorithm

Step 1 : Utilizing the concept of ellipse (Keller, 1989), *k* number of ellipses is drawn taking the depot node and destination node as two foci of each ellipse and time limit, $T^k \times A^k$, as the length of major axis. Then *K* number of paths is initialized by connecting the depot node to each destination node.

Step 2 : Evaluate priority function vectors for all unrouted customer nodes and then select a customer node with the largest priority function value. In case of tie, the node with larger reward is selected. Node is selected arbitrarily in case of the second tie.

Step 3 : Insert it in the current partial path of the corresponding vehicle where it has the minimum insertion time.

Step 4 : Repeat the steps 2 to 3 until it is not possible to include any unrouted customer node to the paths.

End;

4. Numerical Experiment

We examined the effectiveness of the proposed heuristic algorithm through solving two sets of test problems, small size and large size ones, on P.C. with an AMD Athlon (tm) 64 Processor (1.81GHz, 1.00GB). The programming language was Java for the priority algorithm and C++ for pGA of Hwang, Park and Gen (2006). CPLEX 9.0 was utilized to find the optimal solution. Also, the following parameter values were adopted for the heuristic algorithm: 2%, 4%, ···, 20% for r and 0.5, 1.0, 2.0, 3.0 for α, β and γ, respectively. The pGA parameters were set as: crossover probability = 0.7; mutation probability = 0.3; population size = 50. The maximum number of generation (maxGen) was 50 for the small size problems and 250 for the larger size problems. We compared the performances of the heuristics with pGA and CPLEX for the small size problems. For large size problems CPLEX failed to generate an optimal solution within a reasonable computational time and thus the heuristics was compared only with pGA.

4.1 Generation of Test Problems

In small size problems the number of customers ranges from 10 to 15 and the number of employees from 2 to 3. A total of 45 instances were developed with the following parameter values :

- Reward (r_i) is randomly generated using a uniform distribution in [5, 25] (integer)
- Overtime (A_j) of employee is randomly chosen from [20, 100] (integer)
- The coordinates of customer node and employee node are chosen from the interval [-25, 25] with uniform distribution (integer) with the coordinates of depot node being (0, 0).
- Volume (V_i) of items to be delivered to customers is chosen from uniform [10, 30] (integer)
- Volume capacity (C^k) of each vehicle is randomly generated from [50, 100] (integer)

In large size problems the number of customers is either 50 or 100 with the number of employees being 3, 6 or 9. A total of 30 instances were generated with the following parameters :

- Reward (r_i) of customer nodes, volume (V_i) of items to be delivered to customers, overtime (A_j) of employees, and the coordinates of customer nodes and employee nodes are found in the same way as in small sized problems.
- Volume capacity (C^k) of each vehicle is randomly chosen from [200, 400] (integer).

4.2 Computational Results

Let H1 and H2 denote the heuristic algorithms with the priority function of equation (14) and (15), respectively. <Table 1> shows the test results of the small size problems. It lists the problem number and performances of H1, H2, pGA and CPLEX. The first two digits in the problem number correspond to the number of customer nodes,

and the next two digits and last two digits imply the number of employees and the number of replication, respectively. For instance, the problem number 100203 implies that it is the third replication with 10 customers and 2 employees. To measure the relative performance of the algorithms, we introduce *GAP* which is defines as

$$GAP = \frac{Z^* - Z}{Z^*} \times 100(\%)$$

where Z^* and Z is the objective function value obtained by CPLEX and other solution method, respectively. For each solution method, the table shows the objective function value, GAP, and the computation time required (sec.). Only for small size problems CPLEX could generate an optimal solution. It can be observed that the average GAP is 11.2% for H1, 11.4% for H2, and 2.91% for pGA while the average running time of H1, H2, pGA and CPLEX are 0.27 sec., 0.60 sec., 6.00 sec. and 6090.00 sec., respectively. As the problem size becomes bigger, the computation time required of CPLEX increases exponentially. In many cases, CPLEX failed to generate an optimal solution within 24 computation hours, which is denoted by '-' in the objective function value.

Table 1. Computational results for small size problems

Problem	H1			H2			pGA			CPLEX	
	Obj. value	GAP (%)	time	Obj. value	GAP (%)	Time	Obj. value	GAP (%)	time	Obj. value	time
100201	130	3.7	0.24	130	3.7	0.48	132	2.22	5.35	135	260
100202	112	9.68	0.14	112	9.68	0.29	112	9.68	4.78	124	28.75
100203	39	26.42	0.07	39	26.42	0.12	53	0	2.68	53	2.34
100204	108	10.74	0.12	108	10.74	0.22	121	0	2.90	121	9.83
100205	159	0	0.23	159	0	0.46	159	0	5.01	159	1.58
Average	95.4	10.11	0.16	109.6	10.11	0.314	115.4	2.53	4.14	118.4	60.5
110201	115	11.54	0.16	115	11.54	0.37	119	8.46	5.66	130	774.27
110202	182	0	0.29	182	0	0.67	182	0	6.41	182	232.61
110203	158	5.39	0.3	158	5.39	0.71	158	5.39	5.46	167	20.61
110204	114	8.8	0.22	116	7.2	0.48	120	4	5.26	125	3189.44
110205	136	7.48	0.2	136	7.48	0.44	139	5.44	5.75	147	637.36
Average	141	6.64	0.234	141.4	6.32	0.534	143.6	4.39	5.71	150.2	970.86
120201	190	0	0.31	190	0	0.65	190	0	6.54	190	1038.39
120202	104	16.8	0.21	104	16.8	0.51	125	0	5.34	125	263.01
120203	137	12.18	0.22	137	12.18	0.47	140	10.26	6.69	156	2018.31
120204	174	11.68	0.32	174	11.68	0.75	187	5.08	6.82	197	16354.5
120205	152	11.11	0.21	152	11.11	0.53	171	0	6.24	171	893.22
Average	151.4	10.35	0.254	151.4	10.35	0.582	162.6	3.1	6.33	167.8	4113.49
130201	173	0	0.36	173	0	0.87	173	0	7.56	173	46527.1
130202	105	8.7	0.16	105	8.7	0.38	115	0	4.92	115	764.953
130203	98	11.71	0.25	95	14.41	0.56	99	10.81	6.36	111	20082.3
130204	106	-	0.22	106	-	0.47	115	-	6.95	-	-
130205	169	6.11	0.3	165	8.33	0.61	177	1.67	6.60	180	1987.3
Average	130.2	6.63	0.26	125.8	7.86	0.58	136.8	3.11	6.48	144.75	17340.41
140201	131	-	0.28	132	-	0.65	151	-	7.15	-	-
140202	146	-	0.41	143	-	0.97	151	-	8.49	-	-
140203	165	-	0.33	165	-	0.83	158	-	7.7	-	-
140204	128	-	0.34	130	-	0.79	141	-	7.54	-	-
140205	119	13.14	0.22	119	13.14	0.48	136	0.73	6.39	137	22615.3

Average	131.8	-	0.32	131.8	-	0.74	147.4	-	7.47	-	-
150201	85	-	0.16	85	-	0.32	143	-	5.78	-	-
150202	135	-	0.25	135	-	0.6	189	-	8.30	-	-
150203	129	-	0.28	129	-	0.59	133	-	7.11	-	-
150204	213	-	0.54	214	-	1.21	223	-	8.78	-	-
150205	173	-	0.4	176	-	0.87	176	-	8.14	-	-
Average	147	-	0.32	147.8	-	0.72	172.8	-	7.62	-	-
100301	122	31.84	0.23	122	31.84	0.46	179	0	5.09	179	150.33
100302	116	12.78	0.23	116	12.78	0.5	133	0	4.46	133	1235.28
100303	150	11.76	0.3	150	11.76	0.62	170	0	4.89	170	85.64
100304	98	15.52	0.18	98	15.52	0.4	107	7.76	4.22	116	134.8
100305	148	8.64	0.23	148	8.64	0.52	157	3.09	5.67	162	14992
Average	126.8	16.11	0.23	126.8	16.11	0.5	149.2	2.17	4.87	152	3319.61
110301	81	33.61	0.12	81	33.61	0.26	122	0	4.82	122	1231.07
110302	123	13.38	0.24	124	12.68	0.5	142	0	5.34	142	4703.6
110303	203	0	0.39	203	0	0.89	203	0	4.73	203	328.81
110304	167	2.91	0.31	167	2.91	0.75	172	0	5.88	172	28.47
110305	94	30.37	0.16	91	32.59	0.39	118	12.59	5.43	135	3452.08
Average	133.6	16.05	0.24	133.2	16.36	0.56	151.4	2.52	5.24	154.8	1948.81
120301	164	-	0.44	164	-	0.97	164	-	6.38	-	-
120302	165	-	0.34	165	-	0.81	171	-	6.58	-	-
120303	178	-	0.49	178	-	1.09	178	-	5.02	-	-
120304	183	-	0.41	183	-	0.83	190	-	6.84	-	-
120305	152	-	0.47	152	-	1.05	152	-	5.42	-	-
Average	168.4	-	0.43	168.4	-	0.95	171	-	6.05	-	-

<Table 2> shows the computational results for large size problems. Due to the unavailability of optimal solution, H1 and H2 are compared only with pGA. We observe that the proposed heuristics generate satisfactory results. Generally, H2 outperforms H1 and in several cases H2 gives better results than pGA.

Table 2. Computational results for large size problems

Problem	H1		H2		pGA	
	Obj. value	time	Obj. value	time	Obj. value	time
500301	560	19.52	579	22.87	579	293.63
500302	438	14.41	438	15.71	434	230.73
500303	594	22.92	599	26.91	612	320.95
500304	501	17.27	528	20.67	513	329.72
500305	578	22.07	560	22.67	569	292.23
Average	534.2	19.24	540.8	21.77	541.4	293.45
500601	810	49.12	810	54.67	810	352.13
500602	599	30.29	611	35.24	618	305.71
500603	719	46.93	707	52.35	702	335.7
500604	737	59.58	737	68.21	737	334.5
500605	534	25.36	553	30.38	656	262.93
Average	679.8	42.26	683.6	48.17	704.6	318.19
500901	734	72.13	734	83.9	734	319.97
500902	810	71.21	810	81.97	810	277.03
500903	760	64.49	760	73.7	760	301.3
500904	798	65.57	798	75.73	798	340.4
500905	730	63.91	730	73.22	730	290.73
Average	766.4	67.46	766.4	77.7	766.4	305.89
1000301	799	128.91	869	156.61	869	1373.35
1000302	837	127.91	861	146.86	861	1220.92
1000303	843	147.27	901	164.99	943	1396.51
1000304	759	110.28	777	129.25	655	1248.52
1000305	733	93.33	746	98.79	640	1047.8
Average	794.2	121.54	830.8	139.3	793.6	1257.42
1000601	1085	263.34	1082	279.49	915	1306.96
1000602	691	93.1	697	108.97	954	1436.6
1000603	1274	283.04	1314	326.68	1385	1837.18
1000604	1029	205.89	1097	235.27	1109	1343.59
1000605	1078	224.27	1097	261.47	947	1343.59
Average	1031.4	213.93	1057.4	242.37	1062	1453.39
1000901	1379	325.55	1355	367.16	1326	1503.1
1000902	1520	496.97	1520	562.55	1480	1784.71
1000903	1469	438.73	1495	450.66	1532	1656.84
1000904	1509	387.99	1533	524.28	1533	1599.37
1000905	1484	450	1492	501.95	1502	1673.15
Average	1472.2	419.85	1479	481.32	1474.6	1643.43

5. Conclusions

During busy periods of department stores, workers of the handling department sometimes get support from colleagues of other departments to satisfy the delivery requests of the customers. We formulate the above business practice into the mathematical model considering the capacity constraint of vehicle. The problem is found to be NP-hard and thus a heuristic algorithm of construction type is proposed to solve the model. To show the validity of the algorithm, we solve various test problems and make the comparison study on solutions obtained from CPLEX and a genetic algorithm. The results show that the proposed heuristic shows satisfactory results taking far less computation time compared to the others. As a further study, integration of time window in customer node can be suggested.

References

Butt, S. E. and Cavalier, T. M. (1994), A heuristic for the multiple tour maximum collection problem, *Computers and Operations Research*, **21**, 101-111.

Butt, S. E. and Ryan, D. M. (1999), An optimal solution procedure for the multiple tour maximum collection problem using column generation, *Computers and Operations Research*, **26**, 427-441.

Chao, I., Golden, B. L., and Wasil, E. A. (1996), The team orienteering problem, *European Journal of Operational Research*, **88**, 464-474.

Chao, I., Golden, B. L., and Wasil, E. A. (1996), A fast and effective heuristic for the orienteering problem, *European Journal of Operational Research*, **88**, 475-489.

Deitch, R. and Ladany, S. P. (2000), The one-period bus touring problem: Solved by an effective heuristic for the orienteering tour problem and improvement algorithm, *European Journal of Operational Research*, **127**, 69-77.

Fischetti, M., Gonzalez, J. J. S., and Toth, P. (1998), Solving the orienteering problem through branch-and-cut, *INFORMS Journal on Computing*, **10**, 133-148.

Golden, B. L., Levy, L., and Vohra, R. (1987), The orienteering problem, *Naval Research Logistics*, **34**, 307-318.

Golden, B. L., Wang, Q., and Liu, L. (1988), A multifaceted heuristic for the orienteering problem, *Naval Research Logistics*, **35**, 359-366.

Hayes, M. and Norman, J. M. (1984), Dynamic programming in orienteering: Route choice and the siting of controls, *Journal of the Operational Research Society*, **35**, 791-796.

Hwang, H., Park, G. A., and Gen, M. (2006), A priority based genetic algorithm for a variant of orienteering problem, *International Journal of Logistics and SCM Systems*, **1**(1), 34-40

Kataoka, S. and Morito, S. (1988), An algorithm for single constraint maximum collection problem, *Journal of Operations Research Society of Japan*, **31**, 515-530.

Keller, C. P. (1989), Algorithms to solve the orienteering problem : A comparison, *European Journal of Operational Research*, **41**, 224-231.

Keller, C. P. and Goodchild, M. (1988), The multi-objective vending problem : A generalization of the traveling salesman problem, *Environment and Planning B: Planning and Design*, **15**, 447-460.

Laporte, G. and Martello, S. (1990), The selective traveling salesman problem, *Discrete Applied Mathematics*, **26**, 193-207.

Leifer, A. C. and Rosenwein, M. B. (1994), Strong linear programming relaxations for the orienteering problem, *European Journal of Operational Research*, **73**, 517-523.

Liang, Y. C., Kulturel-Konak, S., and Smith, A. E. (2002) Meta heuristic for the orienteering problem, *Proceedings of the 2002 Congress on Evolutionary Computation*, 384-389.

Mocholi, J. A., Jaen, J., and Canos, J. H. (2005), A grid ant colony algorithm for the orienteering problem, *The 2005 IEEE Congress on Evolutionary Computation*, **1**, 942-949.

Ramesh, R., Yoon, Y., and Karwan, M. H. (1992), An optimal algorithm for the orienteering tour problem, *ORSA Journal on Computing*, **4**, 155- 165.

Tang, H. and Miller-Hooks, E. (2005), A Tabu search heuristic for the team orienteering problem, *Computers and Operations Research*, **32**(6), 1379- 1407.

Tasgetiren, M. F. and Smith, A. E. (2000), A genetic algorithm for the orienteering problem, *Proceedings of the 2000 Congress on Evolutionary Computation*, **2**, 910-915.

Tsilligirides, T. (1984), Heuristic methods applied to orienteering, *Journal of the Operational Research Society*, **35**, 797-809.

Wren, A. and Holiday, A. (1972), Computer scheduling of vehicles from one or more depots to a number of delivery points, *Operational Research Quarterly*, **23**, 333-344.