administrative sciences

MDPI

Article

Methodology to Solve the Combination of the Generalized Assignment Problem and the Vehicle Routing Problem: A Case Study in Drug and Medical Instrument Sales and Service

Malichan Thongkham [1,*] and Sasitorn Kaewman [2]

1 Department of Marketing, Mahasarakarm Business School, Mahasarakham University, Maha Sarakham 44000, Thailand
2 Department of Computer Science, Faculty of informatics, Mahasarakham University, Maha Sarakham 44000, Thailand; sasitorn.k@msu.ac.th
* Correspondence: malichan.t@mbs.msu.ac.th

Received: 30 September 2018; Accepted: 20 December 2018; Published: 26 December 2018

Abstract: This article presents algorithms for solving a special case of the vehicle routing problem (VRP). We define our proposed problem of a special VRP case as a combination of two hard problems: the generalized assignment and the vehicle routing problem. The different evolution (DE) algorithm is used to solve the problem. The recombination process of the original DE is modified by adding two more sets of vectors—best vector and random vector—and using two other sets—target vector and trial vector. The linear probability formula is proposed to potentially use one out of the four sets of vectors. This is called the modified DE (MDE) algorithm. Two local searches are integrated into the MDE algorithm: exchange and insert. These procedures create a DE and MDE that use (1) no local search techniques, (2) two local search techniques, (3) only the exchange procedure, and (4) only the insert procedure. This generates four DE algorithms and four MDE algorithms. The proposed methods are tested with 15 tested instances and one case study. The current procedure is compared with all proposed heuristics. The computational result shows that, in the case study, the best DE algorithm (DE-4) has a 1.6% better solution than that of the current practice, whereas the MDE algorithm is 8.2% better. The MDE algorithm that uses the same local search as the DE algorithms generates a maximum 5.814% better solution than that of the DE algorithms.

Keywords: vehicle routing problem; assignment problem; differential evolution algorithm; local search; insertion; drug and medical instrument sales and service

1. Introduction

Marketing techniques are a major factor that can affect sale volume due to the high competitiveness in the real global market. Many techniques have been used in various types of markets to sell different products. Drug and medical instrument sales and service (DMIS) are products that require special marketing techniques. The marketing technique used in DMIS is one where the company sends a seller to meet the customers in their working place to entice, convince, and provide new information to the customers. This can be interpreted as direct sales as the seller attends the customer's location to sell the product. The DMIS customers include hospitals, medical clinics, beauty clinics, drug stores, etc. The profit from DMIS is the number of products the company can sell and minus it the cost incurred. The cost of DMIS comprises production, labor, travel, administrative, accommodation, and managerial costs etc. In this study, the production, administrative, overhead, and managerial costs are approximated as the operation cost which is shown in cost per unit of the product and we assume that other cost terms are not significant and are included in the overhead cost. A certain number of sellers will be sent to sell and service a group of customers. In assigning the different customers to the seller, the sales may change as sellers have different skill levels (experience) and the customers have different stimulation probability (probability of the customer increasing the volume of product when the sellers convince them). When assigning a number of customers to be served by one seller, there is the potential for a generalized assignment problem (GAP). This may occur when the seller constructs their travel plan. The road that connects the customers has different conditions, which affect the driving speed. The driving speed affects both the fuel consumption and the traveling time. This problem can be interpreted as a GAP because we assign the customers to a group of sellers and the traveling times are affected. In one day, the sellers have only eight hours of traveling. Therefore, this problem can also be categorized as a GAP.

The GAP problem is one of how to assign the customers to sellers. When assigning the different customers to the sellers, the traveling route, which has different traveling times (consuming different resources), is affected. Assigning a customer to be served by different sellers may generate different sales volumes, which affects the total profit obtained by the firm. Generally, in a GAP, only the resource consumption difference, when assigning different customers to a seller, is determined but in the proposed problem, the objective function is also affected. Therefore, the proposed heuristic is a special case of GAP.

Many studies have attempted to find a good solution to the traditional GAP. Several exact methods have been used, such as branch and bound (Ross and Soland 1975), the multiplier adjustment method (Fisher and Jaikumar 1981), and the branch and price method (Savelsbergh 1997). Heuristic and metaheuristic methods have been widely used to solve GAP due to requiring much less computational time than the exact methods, such as the differential evolution algorithm (Sethanan and Pitakaso 2016a), Simulated Annealing (Osman 1995), Tabu Search (Laguna et al. 1995; Dıaz and Fernández 2001), genetic algorithm (Chu and Beasley 1997), and bee algorithm (Özbakir et al. 2010).

After the customers have been assigned to a group of sellers, the sellers in that group create their traveling plan. They have to visit all assigned customers within five days, and some customers may require more than one visit during the trip. The sellers may sleep in the city where the last customer in their daily plan is located. This is interpreted as a vehicle routing problem (VRP). The goal of the VRP is to construct a route to minimize traveling distance. The road conditions that affect the driving speed are not considered traditionally. The driving speed affects the fuel consumption rate, which affects the total cost to visit all customers. In this study, the objective function was changed from minimizing the total distance to minimizing the cost to the firm to visit all customers.

VRP involves a defined number of customers and sellers. The solution to the VRP is the optimal traveling plan for each seller. After the seller finishes their route, they return to the depot or the head office. VRP was introduced by Dantzig and Ramser (Dantzig and Ramser 1959), and Lenstra and Kan (Lenstra and Kan 1981) proved that VRP is a Non Polynomial-hard problem (NP-hard problem). Various types of VRP have been consecutively proposed after Dantzig and Ramser (Dantzig and Ramser 1959), such as VRP with time windows, heterogeneous fleet, or Multi-Depot Heterogeneous Vehicle Routing Problem with Time Windows (Bettinelli et al. 2011; Xu et al. 2012). Braekers et al. (2015) provided an overview of the scenario and the problem's physical characteristics, extending the basic incapacitated VRP, which was most often considered in the reviewed articles. The special case of the VRP problem was proposed by Kaewman and Akararungruangkul (2018) and Akararungruangkul and Kaewman (2018), where one pickup point can be visited more than once and the road condition is considered as the objective function.

The proposed problem is one where the GAP and VRP are integrated. To the best of our knowledge, no study has addressed this problem. The VRP is usually integrated with the scheduling problem (Han et al. 2017; Zhan et al. 2015; Nair et al. 2018). The most relevant work was proposed by Hervert-Escobar et al. (2016), who presented an integrated approach to the assignment, scheduling, and routing problems in a sales territory business plan. This paper determined the minimum number of sellers required to attend a set of customers located in a certain region, considering the weekly schedule plan of the visits and forming the optimal route. In our study, the customers are not yet grouped, and the sellers work in a group in which another seller can visit a customer instead of the usual seller when the usual seller is not free. The customers may be visited more than once; they can ask to be visited more than once in a period by a seller. Our routing phase of the problem is also different from that proposed by Hervert-Escobar et al. (2016). The proposed problem integrated the accommodation cost. Therefore, the decision maker must return to the head office as soon as possible so that the accommodation cost is minimized. Furthermore, the sellers' skill level, customers' stimulation level, road condition, and average driving speed, which affects the fuel consumption rate (Fuel Economy in Automobiles (Wikipedia 2018)), were introduced in our problem. Finally, the total profit was used instead of the total cost, as in other studies, as the introduced terms affect the sales volume or the costs. Overall, the problem is a very hard combinatorial optimization problem.

In developing the methodology to solve the VRP and GAP problem, many methods have been proposed, such as Simulated Annealing (Osman 1995), Tabu search (Laguna et al. 1995; Dıaz and Fernández 2001), BEE algorithm (Özbakir et al. 2010), the evolutionary algorithm (Weise et al. 2010), and the genetics algorithm (Liu et al. 2012). Recently, the differential evolution (DE) algorithms have been successfully applied to solve VRP, GAP, and other combinatorial optimization problems.

The DE algorithm applies population-based metaheuristics, which is one of the most powerful and interesting evolutionary algorithms. Generally, it has four common steps: (1) generate the initial solution; (2) the mutation process; (3) the recombination process, and (4) the selection process. DE has been successfully applied in several fields, such as production scheduling (Pitakaso 2015; Pitakaso and Sethanan 2015) and manufacturing problems (López Cruz et al. 2003). Dechampai et al. (2015) proposed a DE to solve the capacitated VRP with the flexibility of mixing pickup and delivery services and maximum duration of a route in the poultry industry. Akararungruangkul and Kaewman (2018) proposed a DE to solve the special case of VRP in a student pickup system. They also modify the mutation process by using a different formulae to increase the search capacity of DE. The computational result showed that the new formula outperforms the traditional DE formula.

Sethanan and Pitakaso (2016b) improved DE by adding two more steps, the reincarnation and survival processes, to improve the intensification search of the DE. These steps were added after the selection process. Sethanan and Pitakaso (2016a) improved the DE algorithms by adding effective local searches to increase the search mechanism of DE for solving the generalized assignment problem. The use of different mutation and recombination processes and matching to improve the solution quality has shown that using different pairs of mutation and recombination processes produced different quality solutions (Sethanan and Pitakaso 2016b; Teoh et al. 2013). From the literature, DE is more effective if the local search is included, but this increases the computational time. DE has been modified by adding some more processes to the original version, such as recombination, reborn, and reincarnation processes, to obtain a better solution. These processes can improve the solution quality of the original DE by enhancing the search capability of the original system.

In this study, we attempted to enhance the search capacity of the DE by adding both diversification and/or exploration and intensification. We modified the recombination process by adding more choice to obtain new trial vectors. Originally, two types of vectors were used: target and mutant vectors. We added two more vectors: (1) best vector, which is used to increase the intensive search capability, and (2) random vector, which is used to increase the exploration capability. The chance of using these four types of vectors is controlled by the newly designed equation.

This article is organized as follows. In Section 2, we explain the case study of the proposed problem and in Section 3 outline our procedure. Section 4 presents the proposed heuristics. Section 5 provides the computational results, and Section 6 outlines our conclusions.

2. The Case Study and Problem Definition

The company we identified is one which sells drugs to hospitals and drug stores in Thailand. The company is located in Bangkok. There are 133 hospitals and 227 drug stores to be serviced. The customers are located in every province of Thailand. The number of sellers that the company hires is 50. The maximum number of sellers in each group cannot exceed 10. Thus, the minimum number of groups has to be at least equal to the number of regions in Thailand. In Thailand, there are five regions which are North, North-east, South, West, and Center-East. Therefore, 50 sellers have to be divided into, at least, five groups. The problem of the company is how to find the daily optimal route that generates the maximum profit for the company. The actors who are involved are customers, sellers, and the cities in which the customer is located. To make the case study are more understandable for the reader, we will use a small example to explain the case study as follows. As shown in Figures 1 and 2, there are 12 customers and three sellers, as an example of the DMIS problem. The minimum demand, the probability of the customer buying more than the minimum demand (stimulation level), the number of visits required, and the order in a particular week are shown in Table 1. Notably, it is possible that a customer needs more than one visit during one planning period. If so, this customer cannot be visited on two consecutive days; at least one day is needed to allow the customer time to consider before meeting the seller again. The unit's selling price is 120 baht, and the total production cost per unit is 40 baht. Therefore, the profit margin of this product is 80 baht per unit.

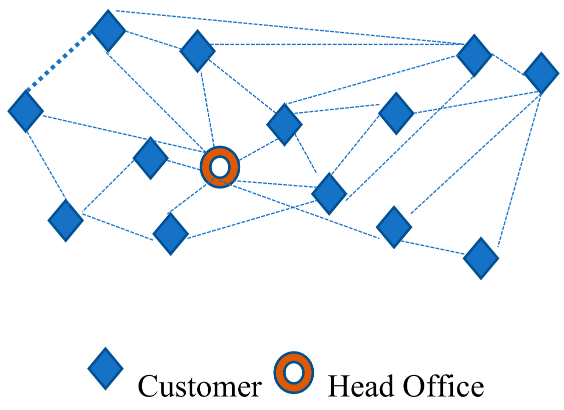

Figure 1. Unsolved problem.

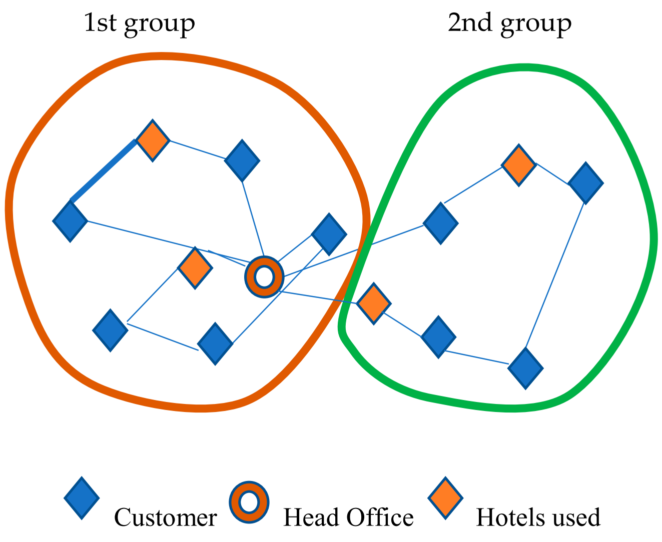

Figure 2. Solved problem.

Table 1. Details of the customers.

#C	#O	#D	Prob	#S	#C	#O	#D	Prob	#S
1	1	200	0.2	240	7	3	300, 320, 310	0.14	150
2	3	150, 120, 130	0.15	110	8	2	110, 210	0	90
3	2	210, 240	0.12	150	9	1	300	0	120
4	1	300	0.2	90	10	1	200	0.21	240
5	1	240	0.24	120	11	1	190	0.13	200
6	2	220, 250	0.32	80	12	1	290	0.05	150

Notes: #C = Customer name, #O = Number of orders placed by a particular customer, #D = Demand of the order, and Prob = probability to increase the demand assumed to be linearly dependent on the seller skill. #S = Service time, which is the direct sell time (minutes).

The distance between customers (road) and the average speed on that road, which depends on the condition of the road, are used to calculate the fuel consumption rate of the road. An example of the proposed problem is as follows. There are three sellers, sellers A, B and C, and the standard salary of a seller is 24,000 Baht; therefore, weekly wages are 6000 baht. The skill levels of A, B, and C are 1.2, 1.0, and 1.3, respectively. The weekly wages of the seller are calculated from the base wages multiplied by the skill level. Therefore, the weekly wage of A, B, and C are 7200, 6000, and 7800 baht, respectively. The required solution of the problem is shown in Table 2.

Table 2. The schedule and route planning of 12 customers and three sellers.

Day	Details	Group 1		Group 2
		A 1.2	**B 1.0**	**C 1.3**
Day 1	Route	1-7	1-2	1-6
	Time used	596	462	265.14
	Order Collect	500	150	220
	Expected Order Collect	98.4	22.5	70.4
	Total Order Collect	488.4	172.5	290.4
Day 2	Route	7-3	2-4-11	6-8-12
	Time used	360	455	454
	Order Collect	210	490	400
	Expected Order Collect	30.24	50.4	18.85
	Total Order Collect	240.24	540.4	418.85
Day 3	Route	3-7-9	11-2	12-6
	Time used	521.25	196.5	202
	Order Collect	620	120	250
	Expected Order Collect	53.76	21.6	104
	Total Order Collect	673.76	141.6	354
Day 4	Route	9-10-3	2-5	6-8-1
	Time used	536.5	234	550
	Order Collect	440	240	210
	Expected Order Collect	84.96	69.12	0
	Total Order Collect	524.96	309.12	210
Day 5	Route	3-2-1	5-7-1	
	Time used	468	496	
	Order Collect	130	310	
	Expected Order Collect	23.4	43.4	
	Total Order Collect	153.4	353.4	

Table 2 provides an example of the expected solution of the DMIS. The 12 customers are divided into 2 groups: the first group includes customers 1, 2, 3, 4, 5, 7, 9, 10, and 11, while the second group includes customers 6, 8, and 12. The first group is serviced by sellers A and B, while the second group of customer is serviced by seller C. The seller starts from customer one's area and returns to customer one as well. In a day, the city in which the last customer in that route is located is used as the rest area (accommodation cost incurred) of the seller before they start the next route the next day.

Table 2 shows that sellers A and B work five days while seller C works four days. The order collected is the minimum order for the customer. The expected order is calculated by the probability of increasing demand of that customer multiplied by the experience level of the seller and the minimum demand of that customer. For example, on day two, seller A has route seven to three, the skill level of seller A is 1.2, and the probability of increasing the demand of customer 3 is 0.12, and demand of customer 3 is 210; therefore, the expected order is 30.24 units. The total product order is the order collected plus the expected order collected. Therefore, on day two, the total product order is 240.24 units. The result of time used, minimum demand collected, and expected increasing demand is shown in Table 2.

The total orders collected (including expected order) of sellers A, B, and C are 2080.76, 1517.02, and 1273.25 units, respectively. Rounded up, these numbers are 2081, 1518, and 1274 units, respectively. Thus, the profit margin of this company will be 4872 × 80 = 389,760 baht. Regarding road travel, the fuel consumed for all trips is 201.474 L. This was determined by the distance between each connection multiplied by the fuel consumption rate for that connection (road). For example, the distance between customers 7 and 3 is 150 km (given information), and the fuel consumption rate of this connection is 0.09. The fuel used to travel this connection is, therefore, 13.5 L. The fuel cost is 33 baht per liter; therefore, the fuel cost is 201.474 × 33 = 6648.642 baht.

The base weekly wage of the seller is 6000 baht, and the income is this base multiplied by the skill level. Thus, sellers A, B, and C have weekly wages of 7200, 6000, and 7800 baht, respectively. Total weekly seller wages are 21,000 Baht.

The next expense of the company is the sellers' commission, which can be calculated from the total sales volume multiplied by the commission rate. If the commission rate is 5% of total sales volume and the total sales are 4872 units, the total income is 4872 × 120 × 0.05 = 29,232 Baht.

The last expense is the hotel cost. The number of nights the sellers have to sleep in a hotel occurs when the last customer that a particular seller visits in a day is not customer 1. From Table 2, there are 11 nights. The hotel cost is 550 Baht per night. Thus, the total hotel cost is 6050 baht.

The total expense comprises fuel cost, weekly wage of the seller, accommodation cost, and commission, and is 6648.642 + 21,000 + 29,232 + 6050 = 62,930.642 Baht. The total profit of the company for this week is 389,760 − 62,930.642 = 326,829.358 Baht.

Figure 3 represents the income component of the company, which is the total sale units (TS) multiplied by the selling price (Pr). The total sale units comprises the minimum sale units, which is the minimum guaranteed order of the customer plus the expected sales units, which is the expected increased order quantity, dependent on the chance to increase the order of the customer and the seller experience. The total cost is subtracted from the income, which includes fuel cost, unit cost, hotel cost, weekly income, and the commission. The total cost component is shown in Figure 4, and the total cost calculation is shown in Figure 5.

Minimum Sale units

+

Expected Sale units

=

Total Sales units (TS)

Total income =TS × Pr

Figure 3. Income component.

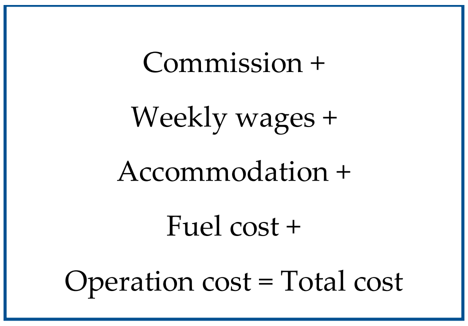

Figure 4. Cost component.

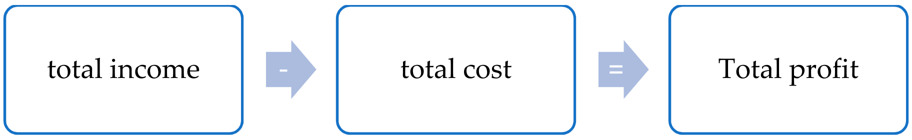

Figure 5. Total profit calculation.

3. Current Practice Procedure (CPP)

The current practice procedure (CPP) is the method that the company is currently using to solve the seller scheduling problem (Figure 1). The CPP procedure is shown in Figure 6.

From the example given in Section 2, the result of applying the CPP method is shown in the following details. From Table 1, there are 19 visits, therefore, B = 19. Then, using Equation (1), the maximum number of visits for each seller is $P = \frac{19}{3} = 6.33 \approx 7$. When B and P are obtained, the next step of the CPP method is to find O^S and O^S. Therefore, O^C = {1, 7, 6, 3, 2, 8, 9, 4, 12, 5, 10, 11}, due to having the following demand, {200, 930, 450, 450, 400, 330, 300, 300, 290, 240, 200, 190}, and O^S = {C, A, B} due to the following skill level {1.3, 1.2, 1.0}. When Step 5 is executed while maintaining conditions 1–5, the result is shown in Table 3.

Table 3. Route results of Step 5.

Day	Seller	C	A	B
1		1-7-6	1-2	1-10
2		6-3-8	2-9-4	10-11-1
3		8-7-6	4-2-5	
4		6-8-3	5-12	
5		3-7-1	12-2-1	

After the daily travel plan of each seller has been determined in Step 5, Step 6 is applied using the criteria of grouping the sellers. Seller B will pick seller C to join their group. The next step is re-routing the grouped sellers using nearest neighbor heuristics. The nearest neighbor is the heuristics where we select the next customer to visit after the current customer by selecting the closest customer to the current customer. The result is shown in Table 4.

Table 4. The new routing for each group of sellers.

	Group 1		Group 2
Day	**C**	**B**	**A**
1	1-7-6	1-8-3	1-2
2	6-10	3-11	2-9-4
3	10-7-8	11-3-1	4-2-5
4	8-6		5-12
5	6-7-1		12-2-1

Step 1: Calculate B, total number of times that all customers need to be visited.

Step 2: Calculate P, maximum visiting times for each seller using following Equation

$$P = \frac{B}{R}$$

when R is total number of seller

Step 3: Find the order of customers O^C which is the order of customers sorted from maximum demand to minimum demand.

Step 4: Find the order of sellers O^S in regards to their skill level from maximum to minimum demand

Step 5: Find the traveling plan for each seller as following way;

5.1 Make free schedule table of 5 working days of each seller

5.2 Assigned the customer into all working days by assign the customer which is in the list O^C to the seller which is in order O^S step by step. In the assigning of the customers to the sellers the following condition must be kept.

Condition 1: The customer needs more than 1 times to be visited, the customer must not be visit on two consecutive days.

Condition 2: Total working time must less than 10 h or 600 min per day.

Condition 3: For each day, the seller will sleep in the last city where the last customer is located.

Condition 4: When we assign one customer to a seller on a particular day, if the next assignment (the closest customer to the last customer in list O^C) means that seller has not enough time, we can pick the next customer in the order O^C to replace it.

Condition 5: When the last customer that a particular seller has to service is visited, the seller must return to depot. The time including the travel back to depot must be less than 600 min.

Step 6: Group the sellers to reduce the working day. The criteria for choosing the sellers to group together is that the seller that has least time used will pick the seller that has greatest time used to join.

Step 7: Re-routing the travelling plan of all available groups is obtained in step (6) using nearest neighborhood heuristics

Step 8: Calculate the total profit (using income and cost component as shown in Figure 3 and 4.

Figure 6. CPP procedure.

The total profit was calculated as shown in Section 2. The result is shown in Table 5. The total profit of this planning is 368,484.523 Baht. We programmed the current practice procedure in C++. In the first iteration of the current practice that we programmed was exactly the same as Steps 1–8. In the next iterations, a random number [0, 1] was picked to multiply the demand of the customer and the skill level of the seller. Therefore, in each iteration, the solution changed because the order of the customer and seller have changed. This enables the current practice to be compared with the proposed heuristics, which will be explained in the next session.

Table 5. Total profit calculation in Thai Baht.

	Income	Expense	Profit
Total income (1)	5068 × 120 = 608,160		
Total production cost (2)		5068 × 40 = 202.720	
Income (1)–(2)			405,440
Commission		5068 × 120 × 0.05 = 30,408	375,032
Weekly wages of Seller		7200 + 6000 + 7800 = 21,000	354,032
Hotel cost		10 × 550 = 5500	348,532
Fuel cost		224.469 × 33 = 7407.477	341,124.523

4. The Proposed Heuristic

Generally, a differential evolution algorithm (DE) comprises four general steps: (1) generate the initial solution; (2) implement the mutation process; (3) execute the recombination process, and (4) complete the selection process. The DE was developed in an attempt to increase the search performance of the original DE algorithm.

The proposed heuristics is called the modified differential evolution algorithm (MDE). Two sets of vectors were introduced to the recombination process: a set of best vectors (BV), and a set of random vectors (RV). Therefore, four sets of vectors were used to process the recombination of the vectors: (1) target vector (TV), (2) mutant vector (MV), (3) best vector, and (4) random vector. To execute the recombination process at each iteration, two values of crossover rate (CR) (*CR*1 and *CR*2) were used to determine which set of vectors would be used. The proposed algorithm is explained step-by-step in the following section.

4.1. Generate Initial Solution

In this section, two main procedures are explained: (1) the encoding and (2) the decoding methods. The encoding method is where we design and present the DE's solution, which is used to execute each step of the DE. As the DE mechanism was first designed to use with continuous optimization, but the proposed problem is a combinatorial optimization, a decoding method was needed to transform the DE solution into the proposed problem's solution.

4.1.1. Encoding Method

A vector that represents the problem comprises two entities: customer's and seller's vector, called target vectors. These two vectors are used to decode to obtain one solution of the proposed problem. The customer vectors are shown in Table 6. There are five customer vectors or number of populations of the customer's vector (NPc). Each vector has the size of 1 × C, where C is number of customers. A customer's vector includes C (in Table 6, C = 12) positions. The value in each position is randomly generated in the first iteration, whereas other iterations are obtained from the DE process.

Table 6. NPc vector of 1 × C.

# NPc	Customer Vectors											
	1	2	3	4	5	6	7	8	9	10	11	12
1	0.92	0.54	0.51	0.22	0.74	0.50	0.98	0.45	0.25	0.90	0.06	0.14
2	0.60	0.32	0.14	0.83	0.89	0.53	0.37	0.87	0.09	0.45	0.87	0.81
3	0.79	0.30	0.15	0.94	0.64	0.17	0.96	0.80	0.53	0.13	0.66	0.51
4	0.90	0.84	0.06	0.48	0.19	0.15	0.72	0.46	0.24	0.14	0.47	0.93
5	0.84	0.93	0.76	0.82	0.33	0.19	0.17	0.40	0.01	0.59	0.23	0.94

Note: # NPc is the vector label, which ranges from 1 to 5.

Another type of vector that was used to construct the solution for the proposed problem is the seller's vector, which is shown in Table 7. A seller's vector has the size of 1 × S and has NP vectors (NP is the number of populations of the seller's vector). This vector is also randomly generated. In Table 7, NPs equals 4 while S = 3.

Table 7. NPs vector of 1 × S.

# NPs	**Seller's Vector**			**# NPs**	**Seller's Vector**		
	1	**2**	**3**		**1**	**2**	**3**
1	0.88	0.45	0.77	3	0.95	0.17	0.65
2	0.61	0.59	0.74	4	0.79	0.53	0.30

Note: #NPs = number of population or number of vectors.

From Table 7, there are three sellers and five vectors. For example, vector 5 includes three positions: 0.65, 0.14, and 0.94. The solution of the proposed problem is determined by the decoding method using these two sets of vectors, which will be explained in Section 4.1.2.

4.1.2. Decoding Method

The decoding method was used to transform the vectors shown in Tables 6 and 7 into the proposed problem's solution, as shown in Figure 7. From the example in Section 2, using the procedure given in Figure 7, G is calculated using Equation (2) when TS = 3 and if MS = 2 when TS = total number of seller, and MS = maximum number of the sellers that can be assigned to each group. Therefore, G equals to 2 groups. Step 2 provides the order of the seller (O^S). This order is obtained by sorting the sellers according to their value in the position of that vector. An example of finding the O^S of vector 2 is shown in Table 8. In Step 3, using the information shown in Table 8, the seller in order O^S is assigned to 2 groups of sellers. When MS = 2, group 1 comprises two sellers (sellers B and C) and group 2 has one member, which is seller A. In Step 4, order O^C is obtained. The sorting is conducted by the value in the position of the vector. An example of the O^C of vector 1 is shown in Table 9. In Step 5, use Equation (3) to calculate Nv. From the example, $Nv = 19/3 = 6.33 \approx 7$. The next step (step 6) constructs the route for each seller. An example of the route construction is shown in Table 10.

Table 8. Results of sorting the seller's vector.

Type of Order	**Seller's Vector #1**			
Original order	seller	A	B	C
	value	0.88	0.45	0.77
Sorted order	seller	B	C	A
	value	0.45	0.77	0.88

Table 9. Results of the calculation of cumulative visiting times.

	Customer's Vector #1											
# Customer	1	2	3	4	5	6	7	8	9	10	11	12
Value in position	0.92	0.54	0.51	0.22	0.74	0.50	0.98	0.45	0.25	0.90	0.06	0.14
# Customer after sorted	11	12	4	9	8	6	3	2	5	10	1	7
Value in position	0.06	0.14	0.22	0.25	0.45	0.50	0.51	0.54	0.74	0.90	0.92	0.98

Step 1: Calculate G which is the number of group of the sellers/customers using the following Equation

$$G = \frac{TS}{MS}$$

When TS = total number of seller, and MS = maximum number of the sellers that can be assigned to each group. Please note that G is an integer. If the result of the division is a real number the largest closest integer is used as *G*

Step 2: Find O^S, the order of the seller obtained by sorting the seller according to their value in position of a seller's vector.

Step 3: Assign the sellers into the each group according to O^S while the maximum number of sellers in each group should not exceed *MS*.

Step 4: Find the order of customers O^C with regard to their value in position of a customer's vector.

Step 5: Calculate *Nv* which is the average number of visits per seller using the following equation

$$Nv = \frac{TV}{NS}$$

When *TV* = total number of visits required by all customers and *NS* is number of seller.

Step 6: Construct the route of each seller in each day during the planning period by using priority. The priority rules are:

6.1. The customer that has the maximum number of visit will be assigned to the first seller in each group first.

6.2. The remaining traveling plan (routing) using the order of O^C

When we construct the route the following conditions need to be satisfied:

Condition 1: Number of times each seller can visit within the planning period should not exceed *Nv*

Condition 2: The construction of the route will execute group by group.

Condition 3: The total time used per day should not exceed predefined limited time.

Step 7: Apply two local search techniques which are (1) exchange algorithm and (2) insertion algorithm

Figure 7. The decoding procedure.

Table 10. Results of Step 4.

		Group 1		**Group 2**
		B 1.0	**C 1.3**	**A 1.2**
Day 1	Route	1-2	1-8-3	1-7
	Time used (minutes)	462	562	596
	Orders Collected	150	330	500
	Expected Orders Collected	22.5	32.76	98.4
	Total Orders Collected	172.5	362.76	598.4
Day 2	Route	2-6-4	3-9-5	7-10
	Time used(minutes)	480.85	343.5	358.287
	Orders Collected	520	540	200
	Expected Orders Collected	130.4	74.88	50.4
	Total Orders Collected	650.4	614.88	250.4

Table 10. *Cont.*

		Group 1		Group 2
		B 1.0	**C 1.3**	**A 1.2**
Day 3	Route	4-2-12	5-3-8	10-7
	Time used	561.5	568.4	193.71
	Orders Collected	410	440	320
	Expected Orders Collected	32.5	70.2	53.76
	Total Orders Collected	442.5	510.2	373.76
Day 4	Route	12-11-6	8-1	-
	Time used	571	115	
	Orders Collected	440	0	
	Expected Orders Collected	104.7	0	
	Total Orders Collected	544.7	0	
Day 5	Route	6-2-1		7-1
	Time used	516		388
	Orders Collected	130		310
	Expected Orders Collected	19.5		52.08
	Total Orders Collected	149.5		382.08

From Table 10, we can observe that we expected to sell a total of 5102.28 ≈ 5103 units. The total fuel used was 206.638 (calculated from the total distance multiplied by the type of road). The cost calculation of the routes constructed from the previous steps is shown in Table 11.

Table 11. Cost calculation of the visiting plan shown in Table 10.

	Income	Expense	Profit
Total income (1)	5103 × 120	612,360	
Total production cost (2)		5103 × 40 = 204,120	
Income (1)–(2)			408,240
Commission		5103 × 120 × 0.05 = 30,618	377,622
Weekly wages of Seller		7200 + 6000 + 7800 = 21,000	356,622
Hotel cost		11 × 550 = 6050	350,472
Fuel cost (Baht)		206.638 × 33 = 6819.054	343,752.946

Note: Unit of income used in this table is Baht.

The last step of the proposed procedure is applying a local search to the solution shown in Table 10. The first local search is the exchange algorithm (EA). EA involves picking two customers and exchanging their position in the route. For example, on day 3, group 1 follows the route 4-2-12 and the next day, 12-11-6. This route consumes 36.413 L of fuel. If customers 2 and 12 are selected to be exchanged, the new route will be 4-12-2 and 2-11-6, and this route consumes 41.9 L of fuel. The fuel consumption was not reduced. Thus, this route is declined. We continued to perform EA until all customers were exchanged. The second local search is the insertion algorithm (IA). This algorithm was used to move a customer from a route into another route such as 4-2-12 and 12-11-6. If customer 2 is moved out from their route to another, for example, between 12-11, then the new routes will be 4-2 and 12-2-11-6. This insertion is needed to check the time limit, and all conditions of the assignment must be satisfied before the move will be permanently inserted. Finally, the insert is only carried out when the fuel usage is reduced.

4.2. Execute Mutation Process

Some target vectors, which are provided in Tables 6 and 7, were randomly selected to execute the mutation process. The result of the mutation process is called the mutant vector. Mutant vectors were produced using Equation (4). We first randomly selected three vectors out of the NPc/NPs target vectors. Customer vectors and seller vectors were treated separately. Let r_1, r_2, and r_3 denote the randomly selected vectors from the target vector (both sets of vectors). F is the scaling factor. In the proposed heuristics all F were set to 0.8; i is the vector number; j is the vector's position; G is the current iteration of the simulation; $X_{r_1,j,G}$, $X_{r_2,j,G}$, and $X_{r_3,j,G}$ are the values in the position of the target vector r_1, r_2, and r_3, respectively; and $V_{i,j,G+1}$ is the mutant vector generated for vector i, position j in iteration $G + 1$.

Each group of vectors (customer or seller) use the same controlled parameters, such as F and CR (recombination rate).

$$V_{i,j,G+1} = X_{r_1,j,G} + F\left(X_{r_2,j,G} - X_{r_3,j,G}\right) \tag{1}$$

An example of how to use Equation (1) is explained as follows. We assume that F = 2. To obtain the first mutant vector we randomly selected three vectors (r_1, r_2, and r_3) which are given in Table 7. The selected target vectors are 4, 2, and 3. The first position of mutant vector #1 ($V_{1,1,G+1}$) is 0.79 + 2 × (0.61 − 0.95) = 0.11. This mechanism will be applied to all position (j = 1, 2, and 3). Although, the mutant vector 1 is 0.11, 1.37, and 0.48. All four mutant vectors (equals to number of NPs) will be executed in the same manner.

4.3. Execute Recombination Process

Originally, Equation (2) is used to construct the trial vector ($U_{i,j,G}$) using the information from the mutant vector ($V_{i,j,G}$) and target vector ($X_{i,j,G}$).

$$U_{i,j,G} = \begin{cases} V_{i,j,G} & if\ rand_{i,j} \leq CR \\ X_{i,j,G} & if\ rand_{i,j} > CR \end{cases} \tag{2}$$

where $rand_{i,j}$ is the random number of position j of vector i.

The MDE in Equation (2) is rewritten as in Equaltion (3)

$$U_{i,j,G} = \begin{cases} V^{BV}_{i,j,G} & if\ rand_{i,j} \leq CR1 \\ V^{RV}_{i,j,G} & if\ CR1 < rand_{i,j} \leq CR2 \\ V_{i,j,G} & if\ CR2 < \ rand_{i,j} \leq CR3 \\ X_{i,j,G} & if\ CR3 < rand_{i,j} \end{cases} \tag{3}$$

where $V^{BV}_{i,j,G}$ is the set of best vectors, $V^{RV}_{i,j,G}$ is the set of random vectors, and $CR1$, $CR2$, and $CR3$ are predefined parameters. $CR1$, $CR2$, and $CR3$ control the moving of the search area of the algorithm. When $CR1$ is high, the solution is guided by the current best solution. If the space between $CR1$ and $CR2$ is high, the new exploration behavior is enhanced. $CR3$ level controls the use of the old target vector, or mutant vector.

The self-adaptive $CR1$, $CR2$, and $CR3$ are used to adjust the parameter values. The adaptive $CR1$, $CR2$, and $CR3$ are calculated as follows. Let $CR2$ be the center probability of selecting all types of vectors. Set minimum $CR2$ (B_{CR}) and maximum $CR2$ (M_{CR}). From the preliminary test, the best B_{CR} is 0.2, and the best M_{CR} is 0.8. $CR2$ is iteratively calculated using Equation (4), where M^T is the maximum iteration, and G is the current iterations.

$$CR2 = \begin{cases} B_{CR} + \frac{G}{M^T} & if\ B_{CR} + \frac{G}{M^T} \leq M_{CR} \\ M_{CR} & otherwise \end{cases} \tag{4}$$

If B_{CR} is 0.2 and M_{CR} is 0.8, $CR3$ is calculated using Equation (5):

$$CR3 = \frac{1.00 - CR2}{2} + \frac{G}{M^T}\left[\frac{M_{CR3} - CR2}{2}\right] \tag{5}$$

M_{CR3} was set to the maximum value of $CR3$, which was 0.9.

The using of best vector (BV) and random vector (RV) is controlled by $CR1$. $CR1$ is calculated using Equation (6):

$$CR1 = \begin{cases} \frac{CR2}{2} - \frac{G^{NB}}{M^T \times 0.25} & if\ \frac{CR2}{2} - \frac{G^{NB}}{M^T \times 0.25} \geq Min^{CR1} \\ Min^{CR1} & otherwise \end{cases} \tag{6}$$

where G^{NB} is the number of iterations when the new best solution is not found and Min^{CR1} is the minimum value of $CR1$ that is allowed. $CR1$ starts from $\frac{CR2}{2}$ and resets itself when the new best solution is found.

4.4. Perform Selection Process

The result of the selection process is the new target vector ($X_{i,j,G+1}$), which uses Equation (7) to select the new target from the previous target vector or the trial vector:

$$X_{i,j,G+1} = \begin{cases} U_{i,j,G} & \text{if } f(U_{i,j,G}) \leq f(X_{i,j,G}) \\ X_{i,j,G} & \text{otherwise} \end{cases} \tag{7}$$

Equation (7) is used to select the best vector between the trial vector and the current target vector to be the new target vectors.

Steps 4.2 to 4.4 are iteratively executed. We concluded the framework of the MDE as a picture as shown in Figure 8.

```
Set NP, CR, F, NP (size of vector)
Generate initial solution
Begin
For $G$ = 1 to $G_{max}$ when $G$ = iterations and $G_{max}$ = Maximum iteration
For N = 1 to NP
Random Generate Target vector $X_{r_1,j,G}$ and RV and update BV
Produce Mutant Vector N (Mutation Process) using the following equaltion.
  $V_{i,j,G} = X_{r1,j,G} + F(X_{r_2,j,G} - X_{r_3,j,G})$
Produce Trial Vector N (Recombination Process) using formula
        • Using the following equaltions.
          $U_{i,j,G} = \begin{cases} V^{BV}_{i,j,G}\ if\ rand_{i,j} \leq CR1 \\ V^{RV}_{i,j,G}\ if\ CR1 < rand_{i,j} \leq CR2 \\ V_{i,j,G}\ if\ CR2 < rand_{i,j} \leq CR3 \\ X_{i,j,G}\ if\ CR3 < rand_{i,j} \end{cases}$
        • Local search (optional)
Produce New Target Vector (selection Process) using formula (10) and update vector B2 and RS
          $X_{i,j,G+1} = \begin{cases} U_{i,j,G}\ if\ f(U_{i,j,G}) \leq f(X_{i,j,G}) \\ X_{i,j,G}\ otherwise \end{cases}$
 Update BV, RV and other parameters as needed.
 End
```

Figure 8. Pseudo code of MDE.

5. Computational Framework and Result

The proposed heuristics was coded in Dev C++ using PC Intel Core i3 CPU 3.70 GHz Ram DDR4 8 GB. Fifteen randomly generated data were tested to compare the proposed heuristics with the current practice procedure that the company was currently using. A real-world case study was solved as well (test instance 16). The details of each test instances are given in Table 12. The detail of the proposed heuristics is shown in Table 13. The simulation was executed five times with different stopping criteria depending on the aim of testing. The best solution out of the five runs was recorded and is reported in Tables 14–18.

Table 12. Characteristics of the test instances and the case study.

# Instances	# Customer	# Seller	# Maxseller	# Instances	# Customer	# Seller	# Maxseller
1	30	4	2	9	150	25	6
2	40	5	2	10	175	25	6
3	45	6	3	11	175	30	8
4	70	8	3	12	190	30	8
5	90	11	3	13	210	30	8
6	120	15	4	14	230	30	10
7	125	16	5	15	230	30	10
8	150	20	5	16	350	50	10

Note: # customer is number of customers, # seller is number of sellers, and # maxSeller is the maximum number of sellers in each group.

From Table 12, the number of customers started at 30 and increased to 230 customers. The case study had 350 customers. The number of sellers ranged from 4 to 50 in the case study, whereas the number of sellers in each group ranged from 2 to 10. We have eight proposed heuristics: DE-1, DE-2, DE-3, DE-4, MDE-1, MDE-2, MDE-3, and MDE-4, the detail of which is shown in Table 13.

Table 13. Details of the proposed heuristics.

Method	Mutation Equation	Recombination Equation	Selection Equation	Local Search	
				Exchange	Insert
DE-1	(1)	(2)	(7)	×	×
DE-2	(1)	(2)	(7)	/	×
DE-3	(1)	(2)	(7)	×	/
DE-4	(1)	(2)	(7)	/	/
MDE-1	(1)	(3)	(7)	×	×
MDE-2	(1)	(3)	(7)	/	×
MDE-3	(1)	(3)	(7)	×	/
MDE-4	(1)	(3)	(7)	/	/

× = not using that method and / = using that method.

The computational results are shown in Table 12, Table 14, and Table 16. The first experiment was executed using the number of iterations as the stopping criteria. BP is the current practice procedure that was programmed in C++. All methods were executed for 1000 iterations each using NP = 10. The computational result is shown in Table 14.

Table 14. Profit and computational time generated by all methods.

		Method								
		BP	**DE-1**	**DE-2**	**DE-3**	**DE-4**	**MDE-1**	**MDE-2**	**MDE-3**	**MDE-4**
1	Profit	89,090	91,245	91,245	91,245	91,245	91,245	91,245	91,245	91,245
	Com.	2.1	5.3	5.6	5.7	5.9	5.9	6.15	6.13	6.3
2	Profit	112,862	113,341	113,891	113,935	113,983	114,143	114,219	114,212	114,674
	Com.	3.4	7.2	7.4	7.4	7.8	7.9	7.94	7.98	8.01
3	Profit	113,084	113,691	114,183	114,418	114,581	115,219	115,354	115,891	115,925
	Com.	4.3	7.54	7.56	7.58	7.65	7.65	7.66	7.68	7.71
4	Profit	348,817	361,893	362,438	361,871	364,441	364,512	366,162	366,134	368,946
	Com.	10.2	11.6	12.6	15.5	15.8	14.3	15.1	15.6	17.8
5	Profit	416,491	417,891	419,482	419,812	419,985	420,812	421,472	422,857	438,674
	Com.	15.5	18.4	19.22	19.58	19.59	18.59	19.69	19.54	19.98
6	Profit	698,185	699,678	711,451	711,439	714,484	714,982	716,258	716,583	719,578
	Com.	16.98	20.14	24.35	24.87	26.78	24.79	28.84	28.98	30.14
7	Profit	704,594	729,871	764,158	764,341	771,846	771,924	772,412	772,312	779,638
	Com.	19.24	25.41	28.82	29.76	32.84	29.57	33.79	33.81	38.47
8	Profit	849,721	852,744	859,769	858,338	859,894	859,913	863,871	863,883	867,911
	Com.	23.78	28.89	34.78	35.91	40.71	34.78	41.41	42.35	47.87
9	Profit	848,871	851,885	853,139	852,984	854,475	854,983	856,862	856,915	861,476
	Com.	24.41	28.41	34.91	36.78	40.96	38.41	41.58	42.93	46.89
10	Profit	913,819	915,776	917,618	917,887	918,198	919,374	920,693	920,581	923,783
	Com.	25.35	30.44	39.86	39.71	43.58	38.71	44.19	44.45	49.17
11	Profit	914,138	915,917	917,831	918,094	919,577	919,682	922,941	922,785	923,718
	Com.	28.98	34.26	41.47	42.73	48.17	43.01	46.85	46.91	50.79
12	Profit	945,722	951,898	963,861	963,985	965,152	965,265	968,779	968,816	969,754
	Com.	30.81	40.14	46.85	47.19	51.24	45.17	51.47	51.34	56.81
13	Profit	961,649	968,984	971,117	971,213	973,381	973,512	974,164	974,247	976,889
	Com.	34.15	44.94	49.81	49.19	54.84	48.48	52.15	52.69	58.49
14	Profit	981,485	984,579	990,141	990,468	992,774	993,671	994,731	994,889	996,154
	Com.	38.97	48.11	51.57	52.14	56.81	53.99	59.71	59.19	67.54
15	Profit	985,841	986,453	994,724	994,813	995,711	995,714	997,619	997,187	998,686
	Com.	39.81	51.34	53.48	53.81	63.14	58.93	67.41	67.01	70.45
16.	Profit	1,431,901	1,488,571	1,494,198	1,494,248	1,501,991	1,501,981	1,580,806	1,581,117	1,600,768
	Com.	54.43	69.98	87.84	88.69	94.51	92.15	95.86	96.11	104.43

Note: Profit = profit generated by the methods (Baht) Com. = computational time (minutes), BP = Current practice method.

From the computational result shown in Table 14, DE with local search always improved the solution quality of the algorithm, as did MDE. To determine if the solution quality was significantly improved, the results of statistical testing are shown in Table 15.

Table 15. Statistical test *p*-value of the Wilcoxon Sign Rank Test of results shown in Table 14.

	BP	**DE-1**	**DE-2**	**DE-3**	**DE-4**	**MDE-1**	**MDE-2**	**MDE-3**
DE-1	$\geq$0.0004							
DE-2	$\geq$0.0004	0.00064						
DE-3	$\geq$0.0004	$\geq$0.00064	0.192					
DE-4	$\geq$0.0004	$\geq$0.0008	$\geq$0.00064	$\geq$0.00064				
MDE-1	$\geq$0.0004	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064			
MDE-2	$\geq$0.0004	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064		
MDE-3	$\geq$0.0004	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	0.28041	
MDE-4	$\geq$0.0004	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064

From Table 15, we can observe that all MDE outperformed all DE algorithms. From the statistical result, either the DE or MDE algorithm, when using exchange or insertion heuristic alone, are not significantly different, which means exchange and insertion heuristic (DE-2–DE-3, MDE-2–MDE-3) have no different performance in finding a good solution. When using exchange and insertion heuristics together, the method outperforms all other heuristics.

From the computational result, MDE-4 is the best algorithm among all proposed heuristics. It generated the maximum profit for all tested instances. Comparing DE and MDE, which used the same local search (DE-2⇔MDE-2, DE-3⇔MDE-3, DE-4⇔MDE-4), MDE outperformed DE by using a new recombination process to improve the quality of the original DE.

Table 14 provides the result of simulation when all proposed heuristics use the same number of iterations (1000 iterations) as the stopping criteria. Table 16 provides the result of the simulation when we set the computational time to 90 min. We aimed to determine the performance of all heuristics when using the same computational time.

Table 16. Profit generated in 90 min using different methods. "Best" Best is the best solution amongst all compared heuristics.

	BP	DE-1	DE-2	DE-3	DE-4	MDE-1	MDE-2	MDE-3	MDE-4	Best
1	90,101	91,245	91,245	91,245	91,245	91,245	91,245	91,245	91,245	91,245
2	113,107	113,391	114,013	114,002	114,110	114,441	114,674	114,674	114,674	114,674
3	113,510	114,108	114,871	114,996	115,110	115,819	116,312	116,236	116,718	116,718
4	351,914	362,110	362,549	362,481	364,912	365,912	367,814	367,711	369,127	369,127
5	416,819	419,111	420,018	420,188	421,018	424,992	428,917	428,291	439,188	439,188
6	698,994	701,118	713,380	713,310	715,006	715,195	718,221	718,127	720,192	720,192
7	709,182	739,819	770,018	771,910	776,581	777,918	778,001	778,248	781,104	781,104
8	850,018	853,191	859,891	858,500	859,918	861,014	867,171	867,143	869,982	869,982
9	849,925	852,091	853,419	853,391	855,168	854,812	856,918	856,147	863,181	863,181
10	913,917	915,981	918,819	918,712	918,718	920,187	921,821	921,817	924,818	924,818
11	914,618	917,192	918,198	918,981	919,896	920,018	922,872	922,994	924,781	924,781
12	945,722	952,487	964,104	964,019	965,941	966,180	968,981	969,917	971,920	971,920
13	963,388	969,017	971,981	971,213	973,307	973,819	975,191	975,028	977,812	977,812
14	982,184	989,817	991,216	991,875	993,100	993,805	994,983	994,903	996,904	996,904
15	985,841	901,014	994,981	994,916	996,053	996,182	998,094	998,295	999,172	999,172
16	1,477,124	1,488,571	1,494,198	1,494,248	1,501,991	1,521,981	1,580,806	1,581,117	1,598,813	1,598,813

The statistical test has been executed using Wilcoxon Sign Rank test with 95% confident interval, and the computational result is shown in Table 17.

Table 17. Statistical test of results shown in Table 16.

	BP	DE-1	DE-2	DE-3	DE-4	MDE-1	MDE-2	MDE-3
DE-1	$\geq$0.00714							
DE-2	$\geq$0.00044	$\geq$0.00064						
DE-3	$\geq$0.00044	$\geq$0.00064	0.9521					
DE-4	$\geq$0.00044	$\geq$0.00064	0.0012	$\geq$0.00064				
MDE-1	$\geq$0.00044	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00318			
MDE-2	$\geq$0.00044	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064		
MDE-3	$\geq$0.00044	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	0.92828	
MDE-4	$\geq$0.00044	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064	$\geq$0.00064

From statistical test shown in Table 17, the results correspond to the results shown in Table 15. All MDE algorithms outperformed the DE algorithms. Two local search procedures, exchange and insertion, were not significantly different in finding a solution, but when used together, they performed well.

Table 18 shows the percent difference in the heuristics, and the best solution found compared to all heuristics.

Table 18. Percent difference in a heuristic compared with the best solution compared to all heuristics.

	BP	DE-1	DE-2	DE-3	DE-4	MDE-1	MDE-2	MDE-3	MDE-4
1	1.27	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	1.39	1.13	0.58	0.59	0.49	0.20	0.00	0.00	0.00
3	2.83	2.29	1.61	1.50	1.40	0.78	0.35	0.41	0.00
4	4.89	1.94	1.81	1.83	1.16	0.88	0.36	0.39	0.00
5	5.37	4.79	4.56	4.52	4.32	3.34	2.39	2.54	0.00
6	3.03	2.72	0.95	0.96	0.73	0.70	0.27	0.29	0.00
7	10.14	5.58	1.44	1.19	0.58	0.41	0.40	0.37	0.00
8	2.35	1.97	1.17	1.34	1.17	1.04	0.32	0.33	0.00
9	1.56	1.30	1.14	1.15	0.94	0.98	0.73	0.82	0.00
10	1.19	0.96	0.65	0.66	0.66	0.50	0.33	0.33	0.00
11	1.11	0.83	0.72	0.63	0.53	0.52	0.21	0.19	0.00
12	2.77	2.04	0.81	0.82	0.62	0.59	0.30	0.21	0.00
13	1.50	0.91	0.60	0.68	0.46	0.41	0.27	0.29	0.00
14	1.50	0.72	0.57	0.51	0.38	0.31	0.19	0.20	0.00
15	1.35	10.89	0.42	0.43	0.31	0.30	0.11	0.09	0.00
16	8.24	7.41	7.00	7.00	6.45	5.05	1.14	1.12	0.00
Ave.	3.16	2.84	1.50	1.49	1.26	1.00	0.46	0.47	0.00

Notes: Ave = average and the % different was calculated using Equation (8).

$$\% \, diff. = \left[\frac{P_B - P_H}{P_B} \right] \times 100\% \tag{8}$$

where P_B denotes the profit generated from the best heuristic compared among all proposed heuristics and P_H denotes the profit generated from an algorithm that is used to compare it with P_B.

Table 18 shows that MDE-4 was the best heuristics among all proposed heuristics due to generating profit 0.00% lower than that of the best solution which means it generated 100% maximum profit among all proposed heuristics.

The next experiment was performed to discover if each heuristics perform the same when using different computational times. The computational time was set to 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, and 120 min and the results are shown in Table 19.

Table 19. Profit of the case study using different controlled simulation times in Thai Baht.

Min	BP	DE-1	DE-2	DE-3	DE-4	MDE-1	MDE-2	MDE-3	MDE-4
10	1,415,328	1,422,845	1,426,545	1,425,747	1,431,583	1,422,283	1,456,237	1,455,892	1,469,835
20	1,422,454	1,424,187	1,432,954	1,431,852	1,444,197	1,459,821	1,458,472	1,454,678	1,488,274
30	1,425,281	1,428,682	1,435,892	1,437,249	1,445,844	1,468,278	1,474,812	1,478,415	1,499,825
40	1,428,690	1,433,741	1,447,296	1,438,713	1,449,818	1,469,194	1,518,721	1,527,294	1,532,857
50	1,430,814	1,439,129	1,448,721	1,441,724	1,459,821	1,470,731	1,522,582	1,534,827	1,558,721
60	1,431,901	1,440,571	1,454,198	1,454,248	1,461,991	1,472,815	1,530,806	1,531,117	1,550,768
70	1,449,924	1,450,872	1,458,819	1,458,787	1,473,877	1,474,761	1,543,984	1,538,732	1,569,821
80	1,442,298	1,451,874	1,457,197	1,461,782	1,478,882	1,478,728	1,557,612	1,544,985	1,569,743
90	1,447,659	1,455,871	1,467,681	1,468,763	1,487,614	1,486,287	1,563,984	1,558,219	1,587,529
100	1,453,376	1,467,916	1,472,644	1,478,779	1,499,817	1,499,921	1,569,821	1,568,873	1,585,872
110	1,467,871	1,478,719	1,489,712	1,488,763	1,499,872	1,512,763	1,578,582	1,578,872	1,592,465
120	1,477,124	1,488,571	1,494,198	1,494,248	1,501,991	1,521,981	1,580,806	1,581,117	1,598,813

From the results in Table 19, we plotted a graph to visualize the progression of the simulation result when using different computational times (Figure 9). From Figure 9, all MDE algorithms found a good solution faster than the original DE algorithms. The algorithms that used local search started with a better solution than the algorithms without local search. The other value that is interesting is the difference between the solution generated from DE and MDE algorithms when using 10 min and 120 min. The large gap in the solution given different computational time means the capability of searching for a new solution is better than the algorithms with a smaller gap. The gap of each method is shown in Figure 10.

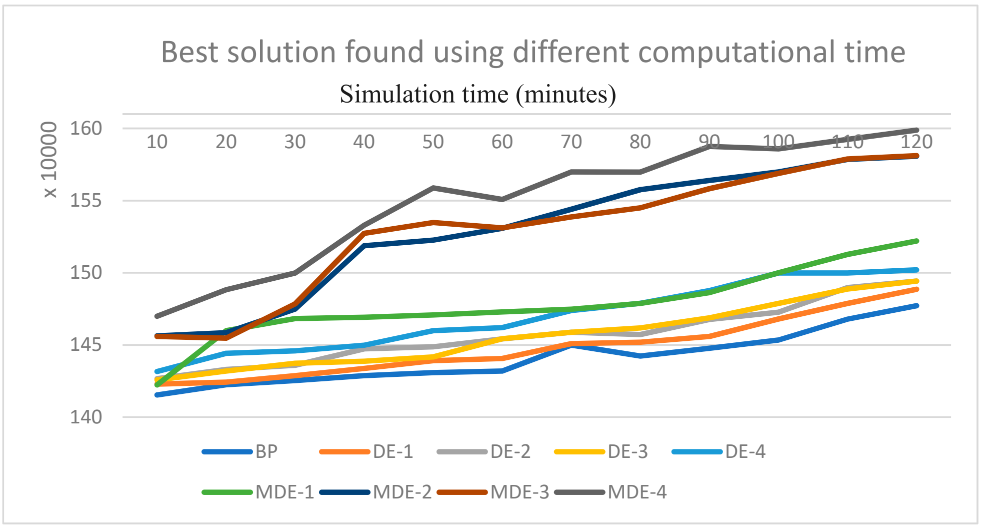

Figure 9. Best solution found using various computational time.

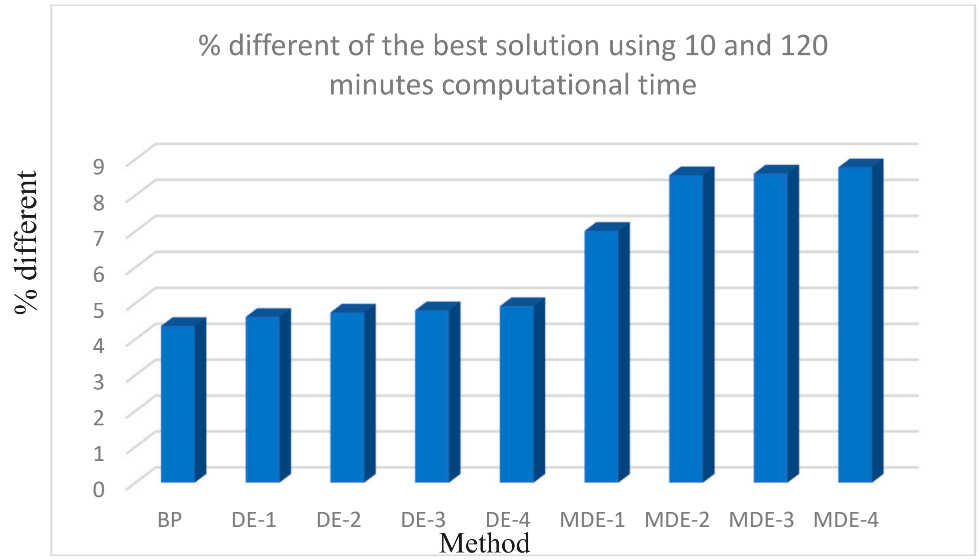

Figure 10. Percent difference in the best solution found using 10 and 120 min of computational time.

From Figure 10, the algorithms with local search had a larger gap between the first and last solution. This means that the local search enhances the searching capability of the algorithm. The MDE algorithms had larger gaps than the DE algorithms. This means the new recombination formula improves the performance of the original DE for finding a better solution.

6. Conclusions and Outlook

This article presented algorithms to solve a special case of the vehicle routing problem. The special case of the VRP had the following attributes, which are different from the original and other variants of VRP: (1) One customer can be visited more than once and if more than one visit is required, the seller cannot visit that customer on two consecutive days. (2) The customers are grouped, and each group includes some sellers. The sellers in the group can share the customers. (3) On each day, the seller sleeps in the last city they visit. (4) All sellers must return the head office when they finish their travel plan. (5) When traveling from one customer to another customer, the road condition controls the average speed on that route, which affects the fuel consumption, which may affect the travel plan.

A different evolution (DE) algorithm was used to solve the problem. The recombination process was modified by adding two more vector sets: best vector (BV) and random vector (RV), to additionally use the set of mutant or target vectors. Linear probability was proposed to enable the use of four sets of vectors. Two local search techniques were added to the modified differential evolution algorithm (MDE).

The computational result showed that the MDE algorithms outperform the DE algorithms due to different recombination process. The new recombination process enhances the intensification capability of the original DE. A good solution was guided by BV, and when it was stuck on the local optimal, the RV helped the heuristics to escape the local optimum. New mechanisms were used while we maintained the excellent search ability of the original DE because we still used the TV and MV in the MDE algorithm.

Two local searches were presented. The computational result shows that DE and MDE algorithms that use local search produce better solutions than algorithms without local search. Local searches that are exchange or insertion algorithms performed no differently in improving the solution quality, but when combined, the search capability was better than using only one local search.

In the case study, the best DE algorithm (DE-4) produced a solution 1.6% better than the current practice procedure, whereas the MDE algorithm solution was 8.2% better from the current practice procedure. Pairwise comparison of the different DE and MDE algorithms that did not use local search, used exchange, used insertion, and used both exchange and insertion algorithms produced values of 2.244%, 5.796%, 5.814%, and 6.446%, respectively, which means the recombination process used in MDE algorithms enhances the capability of the DE algorithms for all local search types.

Overall, we conclude that both local search and the new scheme of recombination work together well and increase the search ability of the original DE. This method may enable the invention of new recombination schemes for the original DE, which can increase the search capacity of the DE. An effective local search should increase the searching ability of the DE as well. These two themes are an interesting area of future research.

Author Contributions: M.T. formulate the problem, gather information, make an experiment, write the article and conclusion; S.K. design the algorithm, programming, write the articles and make the conclusion together with M.T.

Funding: This research is funded from Mahasarakarm University, Thailand.

Conflicts of Interest: The authors declare no conflicts of interest.

References

Akararungruangkul, Raknoi, and Sasitorn Kaewman. 2018. Modified Differential Evolution Algorithm Solving the Special Case of Location Routing Problem. *Mathematical and Computational Applications* 23: 34. [CrossRef]

Bettinelli, Andrea, Alberto Ceselli, and Giovanni Righini. 2011. A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows. *Transportation Research* 19: 723–40. [CrossRef]

Teoh, Boon Ean, S.G. Ponnambalam, and Ganesan Kanagaraj. 2013. Differential evolution algorithm with local search for capacitated vehicle routing problem. *International Journal of Bio-Inspired Computing* 7: 321–42. [CrossRef]

Braekers, Kris, Katrien Ramaekers, and Inneke Van Nieuwenhuyse. 2015. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering* 99: 300–13.

Chu, Paul C., and John E. Beasley. 1997. A genetic algorithm for the generalized assignment problem. *Computers & Operations Research* 24: 17–23.

Dantzig, George B., and John H. Ramser. 1959. The truck dispatching problem. *Management Science* 6: 80–91. [CrossRef]

Dechampai, Darat, Ladda Tanwanichkul, Kanchana Sethanan, and Rapeepan Pitakaso. 2015. A differential evolution algorithm for the capacitated VRP with flexibility of mixing pickup and delivery services and the maximum duration of a route in poultry industry. *Journal of Intelligent Manufacturing* 28: 1357–76. [CrossRef]

Dıaz, Juan A., and Elena Fernández. 2001. A Tabu search heuristic for the generalized assignment problem. *European Journal of Operational Research* 132: 22–38. [CrossRef]

Fisher, Marshall L., and Ramchandran Jaikumar. 1981. A generalized assignment heuristic for vehicle routing. *Networks* 11: 109–24. [CrossRef]

Wikipedia. 2018. Fuel Economy in Automobiles. Wikipedia. Available online: https://en.wikipedia.org/wiki/Fuel_economy_in_automobiles (accessed on 1 September 2018).

Han, Shuihua, Ling Zhao, Kui Chen, Zong-wei Luo, and Deepa Mishra. 2017. Appointment scheduling and routing optimization of attended home delivery system with random customer behavior. *European Journal of Operational Research* 262: 966–80. [CrossRef]

Hervert-Escobar, Laura, Francisco López-Ramos, and Oscar A. Esquivel-Flores. 2016. Research article Open access Integrated Approach to Assignment, Scheduling and Routing Problems in a Sales Territory Business Plan. *Procedia Computer Science* 80: 1887–96. [CrossRef]

Kaewman, Sasitorn, and Raknoi Akararungruangkul. 2018. Heuristics Algorithms for a Heterogeneous Fleets VRP with Excessive Demand for the Vehicle at the Pickup Points, and the Longest Traveling Time Constraint: A Case Study in Prasitsuksa Songkloe, Ubonratchathani Thailand. *Logistics* 2: 15. [CrossRef]

Laguna, Manuel, James P. Kelly, JoséLuis González-Velarde, and Fred Glover. 1995. Tabu search for the multilevel generalized assignment problem. *European Journal of Operational Research* 82: 176–89. [CrossRef]

Lenstra, Jan Karel, and AHG Rinnooy Kan. 1981. Complexity of vehicle routing and scheduling problems. *Networks* 11: 221–27. [CrossRef]

Liu, Linzhong, Haibo Mu, Yubo Song, Haiyan Luo, Xiaojing Li, and Fang Wu. 2012. The equilibrium generalized Assignment problem and genetic algorithm. *Applied Mathematics and Computation* 218: 6526–35. [CrossRef]

López Cruz, I.L., L.G. Van Willigenburg, and G. Van Straten. 2003. Optimal control of nitrate in lettuce by a hybrid approach: Differential evolution and adjustable control weight gradient algorithms. *Computers and Electronics in Agriculture* 40: 179–97. [CrossRef]

Nair, D.J., H. Grzybowska, Y. Fu, and V.V. Dixit. 2018. Scheduling and routing models for food rescue and delivery operations. *Socio-Economic Planning Sciences* 63: 18–32. [CrossRef]

Osman, Ibrahim H. 1995. Heuristics for the generalized assignment problem: simulated annealing and tabu search approaches. *OR Spektrum* 17: 211–25. [CrossRef]

Özbakir, Lale, Adil Baykasoğlu, and Pınar Tapkan. 2010. Bees algorithm for generalized assignment problem. *Applied Mathematics and Computation* 215: 3782–95. [CrossRef]

Pitakaso, Rapeepan. 2015. Differential evolution algorithm for simple assembly line balancing type 1 (SALBP-1). *Journal of Industrial and Production Engineering* 32: 104–14. [CrossRef]

Pitakaso, Rapeepan, and Kanchana Sethanan. 2015. Modified differential evolution algorithm for simple assembly line balancing with a limit on the number of machine types. *Engineering Optimization* 48: 253–71. [CrossRef]

Ross, G. Terry, and Richard M. Soland. 1975. A branch and bound algorithm for the generalized assignment problem. *Mathematical Programming* 8: 91–103. [CrossRef]

Savelsbergh, Martin. 1997. A branch-and-price algorithm for the generalized assignment problem. *Operations Research* 45: 831–41. [CrossRef]

Sethanan, Kanchana, and Rapeepan Pitakaso. 2016a. Improved differential evolution algorithms for solving generalized assignment problem. *Expert Systems with Applications* 45: 450–59. [CrossRef]

Sethanan, Kanchana, and Rapeepan Pitakaso. 2016b. Differential evolution algorithms for scheduling raw milk transportation. *Computers and Electronics in Agriculture* 121: 245–59. [CrossRef]

Weise, Thomas, Alexander Podlich, and Christian Gorldt. 2010. Solving real-world vehicle routing problems with evolutionary algorithms. In *Natural Intelligence for Scheduling, Planning and Packing Problems*. Edited by Raymond Chiong and Sandeep Dhakal. Berlin and Heidelberg: Springer, pp. 29–53.

Xu, Yingcheng, Li Wang, and Yuexiang Yang. 2012. A New Variable Neighborhood Search Algorithm for the Multi Depot Heterogeneous Vehicle Routing Problem with Time Windows. *Electronic Notes in Discrete Mathematics* 39: 289–96. [CrossRef]

Zhan, Yang, Zizhuo Wang, and Guohua Wan. 2015. Home care routing and appointment scheduling with stochastic service durations. *SSRN Electronic Journal*. [CrossRef]