https://doi.org/10.1016/j.cie.2017.05.004

A Multiple Colonies Artificial Bee Colony Algorithm for a Capacitated Vehicle Routing Problem and Re-routing Strategies under Time-Dependent Traffic Congestion

K.K.H. NG[a,*], C.K.M. LEE[a], S.Z. ZHANG[a], Kan WU[b] and William HO[c]

* Corresponding author

[a]Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong, China

Address: Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong. Tel.: +852 3400 3899; Fax: +852 2362 5267

[b]Division of System and Engineering Management, School of Mechanical and Aerospace Engineering, Nanyang Technological University

Address: Nanyang Technological University, 50 Nanyang Avenue, N3.2, 02-51, Singapore. Tel: +65 6790 5584, Fax +65 6792 4062

[c]Management and Marketing, Level 10, The University of Melbourne, 198 Berkeley Street, Victoria 3010, Australia. Tel: +61 903 57560

Email Address: kkh.ng@connect.polyu.hk (K.K.H. NG), ckm.lee@polyu.edu.hk (C.K.M. LEE), shuzhu.zhang@connect.polyu.hk (S.Z. ZHANG), kan626@gmail.com (Kan WU) and William.ho@unimelb.edu.au (William HO)

A Multiple Colonies Artificial Bee Colony Algorithm for a Capacitated Vehicle Routing Problem and Re-routing Strategies under Time-Dependent Traffic Congestion

Abstract

An Online Vehicle Routing Problem is a formation of Capacitated Vehicle Routing Problem with re-routing strategy to resolve the problem of inefficient vehicle routing caused by traffic congestion. A flexible delivery rerouting strategy is proposed, which aims at reducing the risk of late delivery. The method of terminating an exploration in a solution by the original ABC algorithm, when the solution is trapped in local optima, is to abandon the solution after specific tolerance *limits* are set. The phenomenon of local optimal traps will be repeated rapidly after a lengthy recursive process and will eventually result in a low quality solution, with a more complex combinatorial problem when the capability of the exploration is restricted by an inflexible termination criterion. Therefore, this paper proposes a novel scheme using a Multiple Colonies Artificial Bee Colony algorithm. The designs of the outstanding bee selection for colony communication show it to be superior in exploitation. The performance of the proposed algorithm is examined through by Capacitated Vehicle Routing instances and a case study, and the results indicate the potential of using real time information for data-driven vehicle scheduling.

Keyword

Online vehicle routing problem, swarm intelligence, artificial bee colony algorithm, multiple colony strategy

1. Introduction

The emergence of real time traffic data provides a new paradigm to solve Online Vehicle Routing Problem (OVRP), which could exploit this information in a more efficient way through the wireless sensor and visual image. The utilization of real time traffic information helps mitigate the delivery risks, especially in high-density traffic areas. Incorporating real time road surveillance is conducive to avoid heavy traffic congestion, so that the risk of late delivery can be decreased and reduce various uncertainties in the transportation cost and delivery time (Arnott et al., 1991; Kim et al., 2005). Real time vehicle scheduling is a critical component in city logistics, however, previous vehicle scheduling either cannot fully utilize the real time traffic conditions or even ignore it. Therefore, in this research, we focus on the exploitation of real time traffic conditions with the purpose of providing useful and valuable insights into vehicle scheduling with quick response and on time delivery. In addition, less greenhouse emission in urban areas will be produced by reducing the fuel consumption and travelling time during distribution.

1.1. Dynamic Vehicle Routing Problem

The classical Vehicle Routing Problem (VRP) is a combinatorial problem to meet delivery schedules and arrangements for a given number of available trucks, set of customers, and corresponding customer order quantities. In contrast to the classical VRP, the Dynamic Vehicle Routing Problem (DVRP) usually includes the evolution and the quality of input data as shown in Figure 1. In fact, the evolution of the VRP data refers to information that may change over time, while the quality of the VRP data denotes the level of uncertainty to retrieve relevant data (Pillac et al., 2013; Ritzinger et al., 2016). Stochastic VRP (SVRP) problems are identified by some of the known input values as arbitrary data, which are realized before the design of particular routes. The prior routes may change afterwards under uncertain random events. Typical examples of SVRP problems are stochastic customers (Gendreau et al., 1995, 1996; Waters, 1989), stochastic order quantities (Haughton, 2002, 2007; Huang & Lin, 2010; Marinakis, 2015) and stochastic time (Ehmke et al., 2015; Errico et al., 2016; Marinakis, 2015; Zhang et al., 2012). The VRP model with known data, which may change time over during the implementation of routes, is characterized as the OVRP. The configuration of OVRP requires an online or real time approach to retrieve relevant information in continuous delivery processes, and is usually associated with a real time communication system to construct a rearranged route (Chen et al., 2006; Du et al., 2007; Li et al., 2009a, 2009b).

The latest wireless and mobile technology can help to capture the real time information, and real time vehicle routing system architecture has been proposed for minimizing the intervention (Giaglis et al., 2004), reducing distribution risk (Ahmadi-Javid & Seddighi, 2013), and stabilizing the impact of transportation disruption on the overall supply chain performance (Wilson, 2007). The Real time stochastic VRP model is a combination of the OVRP and SVRP models given that some data are uncertain and change over time. Yan et. al. have presented a novel approach to develop a route with real time adjustment under the situation of uncertain demand and travel time (Yan et al., 2013).

		Quality of VRP data	
		Definite data	**Stochastic data**
Evolution of VRP data	**Variables are defined in advance**	Classical VRP Model	Stochastic VRP Model
	Variables change overtime	Online VRP Model	Real Time, Stochastic VRP Model

Fig. 1: The Classification of Vehicle Routing Problems (Ritzinger et al., 2016)

OVRP considers the real time data as references or sources of input to adjust the predefined route (Mavrovouniotis & Yang, 2015). The whole process of OVRP is separated into two stages, as shown in Figure 2. The first stage aims to construct the VRP solution with relevant real time data by the selected algorithm. Then, the route will be executed. Once the truck arrives at the destination area, the real time re-optimal routing will be calculated with consideration of the current traffic factors. The route rearrangement process is repeated until all customers are visited.

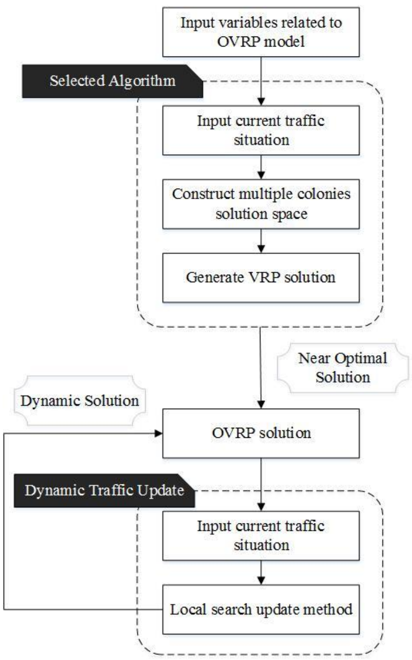


Fig. 2: The Flowchart of OVRP Model

1.2. Swarm Intelligent Algorithms applied in Vehicle Routing Problem

The vehicle routing problem (VRP) is an NP-hard problem and the computational time for solving NP-hard problems increases significantly when the number of nodes increases. It is not feasible to solve the huge routing network problem by an exact algorithm (Bhagade & Puranik, 2012). The complexity of VRP has been widely studied, regarding different objectives, by the proposed heuristic algorithms, like local search method - Tabu Search(Gendreau et al., 1994) and population search based method - Genetic

Algorithm (Baker & Ayechew, 2003) and Swarm Intelligence (SI) (Chen, Hsueh, & Chang, 2006; Zhang et. al., 2014). Swarm intelligence (SI), which is a distributed intelligent paradigm to determine the optimal solution by imitating the collective behavior of decentralized and self-organized natural systems, has been developed in vehicle routing problems, such as Ant Colony Optimization (ACO) techniques (Bell & McMullen, 2004), Particle Swarm Optimization (PSO) (Ai & Kachitvichyanukui, 2009)and Artificial Bee Colony (ABC) algorithms (Szeto et al., 2011). The ABC algorithm was designed by Karaboga (Karaboga, 2005), and has been widely applied to solve the various VRP models and successfully achieved better exploitation and exploration. The computation results conducted by Zhang et al. (2014) denoted that the ABC heuristic and the modified ABC heuristic outperform genetic algorithms in solving Environmental Vehicle Routing Problem (EVRP) problem (Zhang et al., 2014). An enhanced ABC heuristic has been developed by Szeto et al to improve the exploitation ability of the employed bee phase (Szeto et al., 2011).

SI is a collective system among insect or animal behavior with a self-organizing system of interaction of the individuals, which turns in an effort to reach a nearly global optimum. Several update strategies for swarm intelligence have been proposed and integrated with SI to retain a high quality of searching output in various VRP models, which include elitism-based scheme (Mavrovouniotis & Yang, 2014; Yang, 2008), memory-based scheme (Mavrovouniotis & Yang, 2012, 2015; Yang, 2008), random-based scheme (Mavrovouniotis & Yang, 2015). The exchange of individuals is named migration to maintain a restricted level of migration within all colonies, and the elite ants share their knowledge with other colonies. The results from the multiple colonies ACO algorithm are well accepted in searching for the VRP model (Mavrovouniotis & Yang, 2015; Toklu et al., 2014). The elitism-based ABC algorithm and multiple colonies ABC algorithm were introduced a few years ago for numerical function optimization (Mezura-Montes & Velez-Koeppel, 2010; Xiang & Zhou, 2015; Xiang et al., 2015). However, the multiple colonies strategy in ABC algorithm has not been applied to solve the VRP model. Therefore, it is beneficial to study the performance of the multiple colonies ABC with elitism-based selection for solving the VRP model.

In the extant literature, there are few studies that have implemented traffic image processing and OVRP to retrieve the latest traffic information from the surveillance systems in order to design optimal vehicle routing solutions. However, the current road traffic condition does affect the overall distribution time and may have a significant difference from the predefined delivery schedules. The proposed MC-ABC algorithm is proven to be effective and efficient in clarifying real time vehicle routings with the latest traffic information, taking account of the influence of transportation, this attempts to expedite the related research by integrating relevant information, such as road conditions, current traffic and traffic accidents, and real time vehicle routing problems. The contribution of this project significantly affects vehicle routing in a densely populated city, so as to avoid large traffic jams and provide a speedy delivery arrangements for practical and managerial usage in agile logistics.

The rest of this paper is organized as follows. After a brief introduction in the first section, the concepts of vehicle routing and traffic extraction are briefly reviewed in section II. Section III contains the traffic density estimation model and the OVRP model. In section IV, the proposed MC-ABC is examined and evaluated by severed capacitated vehicle routing problem (CVRP) instances. The computational results of the referenced case study are presented in section V. Finally, the conclusions and future work are raised in the last section.

2. Problem formulation

In this research, an OVRP model is proposed to determine optimal vehicle scheduling with the objective of minimizing the total traveling time considering the real time travel conditions. Image processing technologies are employed to estimate the traffic density with regard to the real time traffic conditions. Furthermore, the traffic density estimation acts as an essential external factor affecting the real time vehicle scheduling. The risk of late delivery is alleviated with the incorporation of real time traffic density estimation in the vehicle scheduling.

2.1. Traffic Extraction

In general, the lower traffic density of a road segment implies a faster vehicle speed in the traffic flow. Statistical mapping for identifying the relationship between traffic density and vehicle speed is static. The approximate slope of the relationship between vehicle speed and road density occupation is -1% of the free-flow vehicle speed. A study found that vehicle speed could be doubled when there is a 20% decrease in traffic density (Sen et al., 2013). In our study, the estimated vehicle travelling speed is normalized as a monotone decreasing function to represent the relationship between speed and road occupation in order to determine the current estimated vehicle speed from the traffic density of a road segment.

$$\mathrm{P}(X_t) = \sum_{i=1}^{K} \omega_{i,t} \times \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{1}$$

$$\eta\left(X_t, \mu_{i,t}, \Sigma_{i,t}\right) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} e^{\frac{1}{2}(X_t-\mu_{i,t})\Sigma^{-1}(X_t-\mu_{i,t})} \tag{2}$$

$$\Sigma_{i,t} = \sigma_{i,t}^2 I \tag{3}$$

$$\left|X_t - \mu_{i,t}\right| \leq D\sigma_{i,t-1} \tag{4}$$

$$w_{i,t} = (1-\alpha)w_{i,t-1} + \alpha \tag{5}$$

$$\mu_{i,t} = (1-\rho)\mu_{i,t-1} + \rho I_t \tag{6}$$

$$\sigma_{i,t}^2 = (1-\rho)\sigma_{i,t-1}^2 + \rho(I_t - \mu_{i,t})^2 \tag{7}$$

$$w_{i,t} = (1-\alpha)w_{i,t-1} \tag{8}$$

$$\mathrm{B} = \arg min_b \left(\sum_{j=1}^{b} w_{j,t} > T\right) \tag{9}$$

$$\mathrm{T}_{x,y} = \left|\mathrm{I}_{x,y} - \mathrm{B}_{x,y}\right| \tag{10}$$

$$\mathrm{TDE} = \frac{T_{x,y}}{M_{x,y}} \tag{11}$$

In the method of background subtraction, a Mixture Gaussian background model (GMM) is applied in traffic extraction. Each pixel in the image follows a Gaussian distribution to deal with fluctuation in pixel value, which has been well developed for real time image tracking, especially in the road traffic (Chen et al., 2015; Lee, 2005). The probability of the current pixel value $\mathrm{P}(X_t)$ in formula (1) is measured by the weight of K^{th} Gaussian distribution and the probability density function η at the current pixel. The number of Gaussian distribution is set from i to K. The Gaussian distribution is calculated by formula (2), where $\mu_{i,t}$ is the mean of the K^{th} Gaussian model and $\Sigma_{i,t}$ is the corresponding covariance matrix, given that the matrix is formulated as a three-independent Red-Green-Blue (RGB) channel by formula (3). The new pixel X_t will then be evaluated with each i^{th} Gaussian distribution and find the matching by Formula (4), where D is defined as 2.5. If the matching process is successful, the current Gaussian model is adjusted with the value of the new pixel X_t and update method by formulas (5) to (7). The prior weight $w_{i,t}$ is adjusted by formula (5) with a learning rate α, where α is between 0 and 1. The parameters of mean μ and variance σ of i^{th} Gaussian distribution are also updated by formulas (6) and (7). If the matching is unsuccessful, the prior weight $w_{i,t}$ is then updated by formula (8). The real background B is estimated by Formula (9) with a threshold value T in Figure 3a. In figures 3b and 3c, formula (10) is the subtraction of foreground T between current image I and background image B in absolute value, given that the x-y coordination of the image T, I and B must be the same pixel size. Two morphological operators, which are eroding and dilating operators in OpenCV library, are applied to fine tune the foreground image to connect the central pixel by compass coordinating with surrounding pixels, which allows connection of the broken part of an object. The result of the morphological operation is demonstrated in Figure 3d. The traffic density estimation TDE is computed by a simple density function of formula (11). $M_{x,y}$ is the manual selected road area in figure 3e, while $T_{x,y}$ is the foreground of real traffic in figure 3d.

Fig. 3a: Estimated road background by GMM model

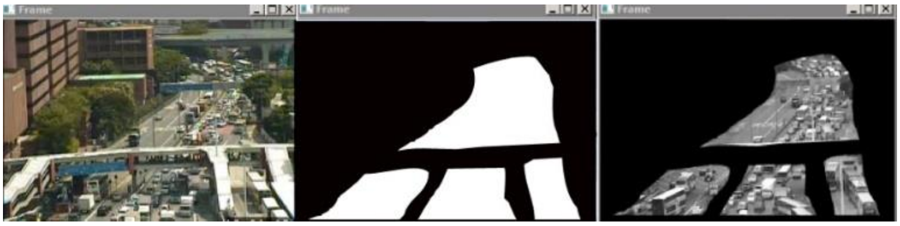


Fig. 3b: Traffic image after filtering and converting into grey-scale model

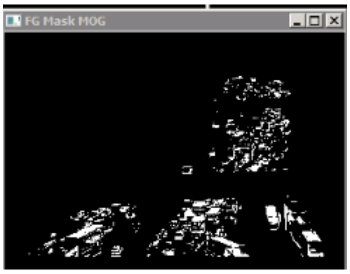


Fig. 3c: Traffic image after threshold operations

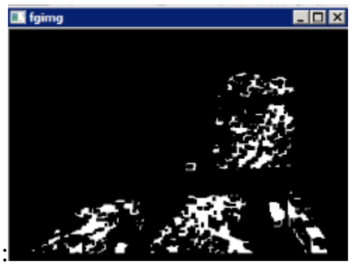


Fig. 3d: Traffic image after morphological operation

Fig. 3e: Interested road region

2.2. Real Time Vehicle Rerouting Problem with Traffic Congestion

The model of VRP is constructed using a graph with customer locations, and the set of edges $A = [(i,j)|i,j \in V, i \neq j|]$ represents the connective path between any two nodes. The settings and notations of VRP are described in Table 1. In this research, a single depot is denoted as node 0, while customers are represented using the remaining nodes from 1 to n. Travelling distance d_{ij} describes the distance between node i and node j, which is further processed with the traffic condition to calculate the travelling time t_{ij}. Traveling time t_{ij} is a real time variable that can change with regard to the traffic condition. Each customer requests a nonnegative demand $r_i(i = 1,2,\dots,n)$, which suggests delivery plans from the depot to all customers. There are maximum m vehicles available. Vehicles are denoted as $k = \{1,2,\dots,m\}$ with the same maximum capacity Q.

Table 1: The Settings and Notations of OVRP

Mathematical Model	Meanings
$G = (V,A)$	The Graph
$V = (0,1,2,\dots,n)$	The set of nodes in the graph
$A = [(i,j)\|i,j \in V, i \neq j\|]$	The set of edges in the graph
$d_{ij}(\forall i,j,i \neq j)$	The travelling distance between *i* and *j*
$r_i(i = 1,2,\dots,n)$	The demand of customer *i*
n	The number of customer
m	The maximum number of vehicle
$k = (1,2,\dots,m)$	The index of vehicle
Q	The vehicle capacity
sp	Vehicle speed with free-flow
TDE	The traffic condition estimation
$t_{ij}(\forall i,j,i \neq j)$	The travelling time between *i* and *j*

Problem Formulation in Each Colony

Decision variables

$$x_{ijk} = \begin{cases} 1, \text{if node } i \text{ is served by vehicle } k \text{ after node } j \\ 0, \text{otherwise} \end{cases} \quad (12)$$

$$y_{ik} = \begin{cases} 1, \text{if node } i \text{ is served by vehicle } k \\ 0, \text{otherwise} \end{cases} \quad (13)$$

Objective Function for OVRP

$$min\, f = \sum_i \sum_j \sum_k t_{ij} * x_{ijk} \qquad (14)$$

Constraint

Vehicle Load:

$$\sum_{i=1}^{n} r_i y_{ik} \leq Q, \forall k \qquad (15)$$

Sub-tour elimination

$$X = (x_{ijk}) \in S, (S \subset V[0]) \qquad (16)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ijk} \leq |S| - 1 \; (|S| \geq 2; i \neq j; \forall k) \qquad (17)$$

Traffic condition, speed, time and distance

$$t_{ij} = \frac{d_{ij}}{sp(1-TDE)}, \forall i, j, i \neq j \qquad (18)$$

Maximum Number of vehicle

$$\sum_{k=1}^{m} x_{0jk} < m, \forall j \qquad (19)$$

Vehicle usage

$$\sum_{k=1}^{m} y_{ik} = 1, \forall i \qquad (20)$$

Customer assignment:

$$\sum_{i=0}^{n} x_{ijk} = y_{jk}, \forall j, i \neq j, \forall k \qquad (21)$$

$$\sum_{j=0}^{n} x_{ijk} = y_{ik}, \forall i, i \neq j, \forall k \qquad (22)$$

Objective function (14) is designed to minimize the total travelling time. Constraint (15) indicates that the vehicle capacity cannot be violated by the vehicle load. Constraints (16) and (17) eliminate possible sub-tours. S is a subset of the vertices of a graph G, where S connects the set of edges having both endpoints in S, with non-trivial cut to perform subtour elimination constraints. Constraint (18) calculates the travelling time taking account of the traffic density estimation (TDE), where TDE if between 0 and 1. There may be spare vehicles for delivery. The number of used vehicles cannot exceed the maximum number under Constraint (19). Each customer can be visited and served by only one vehicle with Constraint (20). Constraints (21) and (22) confirm that the visiting vehicle and the serving vehicle for each customer have to be the same.

3. Methodology

3.1. Proposed Multiple Colonies Artificial Bee Colony Algorithm

The conventional ABC algorithm uses random operators to maintain the diversity within the population, while the modified ABC algorithm in our previous research applies knowledge transfer for every employed bee to achieve a better exploitation ability from a previous iteration (Zhang et al., 2014). Although the modified ABC algorithm shows its significant and distinguishable value in exploitation, the premature convergence can happen and lead to the loss of diversity under an intensive sub-route exchange strategy. In order to balance the ability between exploitation and exploration, a multiple colonies artificial bee colony (MC-ABC) algorithm is proposed. The fundamental concept of multiple colonies is to apply parallel procedures with the programming scheme to uphold the divergence of the population with the random exploitation method, but allow satisfactory information exchange among colonies to avoid premature convergence. The parameter settings of the MC-ABC algorithm are denoted in Table 2. The flow chart of the MC-ABC algorithm is shown in Figure 4. The numbers of solutions are fixed to the half size of the bee colony, i.e. $SN = {CS}/{2}$. In addition, the criterion of termination of neighbor searching and abandoning a solution, called $limit$, is defined as the number of solutions multiplied by the dimension of an individual solution $limit = SN \times Dim$. An extra parameter is introduced here: the number of colonies C. Each of the colonies contains the same number of solutions, and the maximum number of colonies is equivalent to C. Employed bees, and onlooker bees carry out the same neighbor search operation as the original ABC algorithm. The major contribution of the MC-ABC algorithm is to provide elitism-based or random-based information exchange in the scout bee phase, which lead to a better exploitation and exploration of overall ABC performance.

Table 2: Notation for Multi-Colonies Artificial Bee Colony Algorithm

Notations	Explanation
CS	The size of bee colony
SN	The number of solutions
C	The number of colonies
$MaxIterations$	The maximum number of iterations
$limit$	The criterion of termination for neighbor searching, and abandoning a solution
Dim	The dimension of an independent solution
$x_{l,} l = 1,2, \dots, C$	The l^th position of a multi-colonies solution
$x_{li}, i = 1,2, \dots, SN$	The solution representation of an individual food source
$x_{lij}, j \le Dim$	The j^th position in a sequential solution x_i
$fun(x_{li})$	The objective value of solution x_i
$fit(x_{li})$	The fitness value of solution x_i
$Prob_{li}$	The probability of an individual solution being selected in the population
$Cumu\_Prob_{li}$	The cumulated probability of an individual solution in ascending order in the population
$\bar{x}_i$	The neighbor solution of solution x_i
$trial(x_{li})$	The accumulated value in which the quality of a solution x_i cannot be enhanced
ω	Random number, $0 \le \omega \le 1$
φ	Random number, $-1 \le \omega \le 1$

3.1.1. Multiple Colonies Strategy

The multiple colonies strategy is incorporated in the ABC algorithm, which is one of the major elements in constructing the MC-ABC algorithm. The operation in the employed bees phase, onlooker bees phase among all colonies is an independent process, which denotes the same exploitation ability in each colony without any interference during the iterative process by other colonies. Information sharing among colonies is involved when the trial of any solution is greater than the maximum tolerance of neighbour searching. Moreover, a tactic to replace a solution across colonies is introduced in section 3.1.5. The number of colonies is simply defined as 3 for an ease of evaluation. The approach of the Elitism-based Multiple Colonies ABC algorithm (EBMC-ABC) and the Random-based Multiple Colonies ABC algorithm (RBMC-ABC) is to select a solution from other colonies based on elitism-based selection or random-based selection. The selection criteria in elitism-based approach and random-based approach are discussed in section 3.1.5.1 and 3.1.5.2 correspondingly.

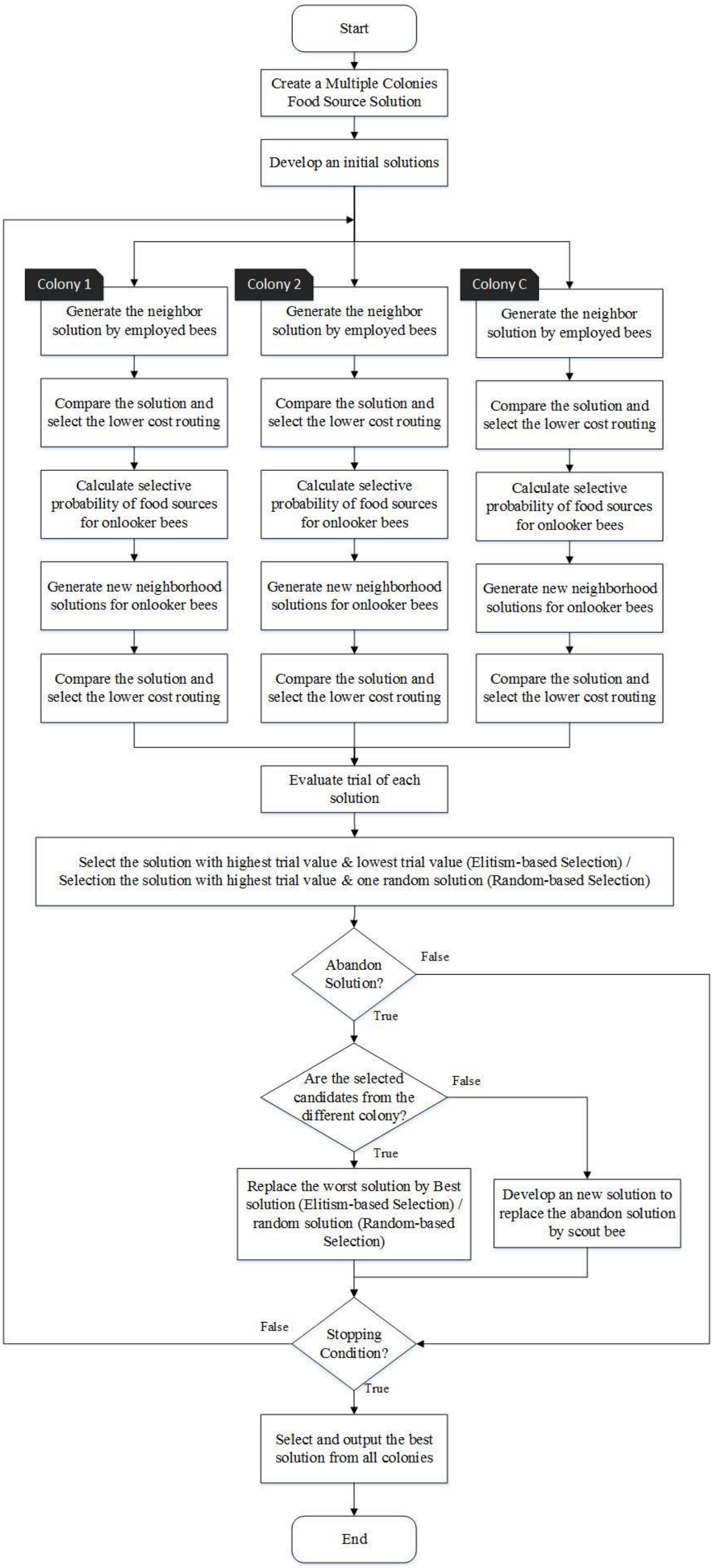


Fig. 4: The Flowchart of Multiple Colonies Artificial Bee Colony Algorithm

3.1.2. Initialization Phase

In the standard ABC algorithm, the initial paths are constructed from continuous variables within a range. However, the dimensional variables are discrete and identical in value to represent the specific customer. With the purpose of facilitating the arrangement of feasible solutions for VRP, each set of solutions is generated randomly as initial food sources. In Figure 5a, each customer will randomly be assigned to the k vehicle route. The capacity of vehicles is considered at the initial stage to provide a feasible solution for each food source. A total of $C \times SN$ initial solutions are randomly constructed by the above arrangement for all bee colonies, where each colony contains a predefined number of solutions SN by the following food source matrix $x_{li}, l = 1,2, \dots, C; i = 1,2, \dots, SN$.

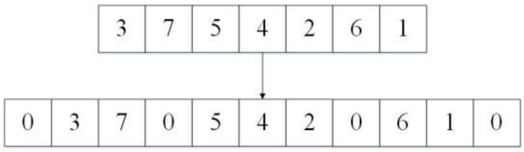


Fig. 5a: The Generation of Initial Solution

In determining the approximate and feasible solution, the random approach on assigning customers to the current vehicle scheduling, with regard to their maximum vehicle load, is then solved iteratively by the MC-ABC algorithm. The randomized solution will then be checked the any violation of the constraints, i.e. maximum load of truck capacity constraint. If any violation of constraint found in the stage, the generation process will be repeated until a feasible solution is found. This approach has the advantage of guaranteeing a feasible solution and facilitating the computation in recursive procedures, as each customer can be placed freely in the dimension space, which is the delivery schedule, under the influence of searching heuristics. In the event that the updated solution violates the VRP constraint, the previous solution will be considered as a current best solution and proceeds to the next stage to guarantee a feasible solution. Once all the food sources in each colony have been generated, the fitness value fit_{li} of each solution is calculated by equation (23). Then, the corresponding termination criterion $trial$ is initialized as zero. $trial$ is the parameter to accumulate the number of unsuccessful updates by the neighbor search operators, and is able to provide an adaptive approach for scout bees to realize the chance of being trapped in local optima of some solutions x_i. In the proposed algorithm, this control parameter becomes an important control parameter in information exchange among all the colonies.

$$fit(x_{li}) = \frac{1}{1+fun_{li}}, \forall l, \forall i \tag{23}$$

3.1.3. Employed Bee Phase

Neighborhood operators of the ABC heuristic, which are performed by employed bees, are used to create a new solution $\bar{x}_i$ from x_i. The set of pre-defined neighborhood operators is applied once for each food source solution. In each neighbor search, the algorithm compares the objective function between the original solution x_i and neighbor solution $\bar{x}_i$. If a better objective value or no violation of any constraints are found after the neighbor operation, the new route will replace the previous one; otherwise, the solution x_i remains unchanged. The crossover operator is still preserved in the MC-ABC algorithm. However, the candidate option of sub-route exchange is restricted within the same colony's solution. According to this arrangement, the diversity of all bee colonies is sustained to prevent the occurrence of premature convergence. In addition, the neighbor search operators include:

(A) Neighbor Search - Swap Operator

This operator selects a random position in each food source to apply the swapping of two sub-sequence on the positions i, j. The length of the selected sub-sequence is a random number from 1 to 2 in our proposed algorithm to enhance the convergence of the neighborhood search (Szeto et al., 2011).

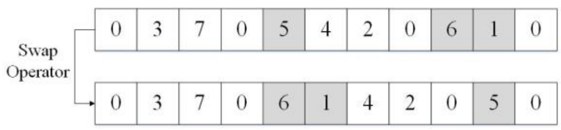


Fig.5b: Swap Operator

(B) Neighbor Search – Insert Operator

This operator chooses positions i, j. Customer i is extracted from position i and inserted in position j.

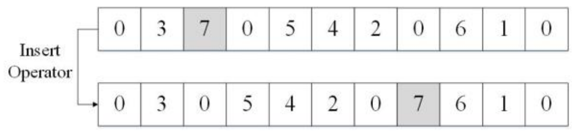


Fig. 5c: Insert Operator

(C) Neighbor Search – Reverse Operator

This operator selects a range at random, which is smaller than the size of the dimension, and applies reverse order to the selected region.

Fig. 5d: Reverse Operator

(D) Neighbor Search – Crossover Operator

This operator selects a sub-route by proportionate reproduction scheme, which is smaller than the size of the dimension, and retains the position of the selected cells. The remained dimension is filled up by another food source in sequence. The selection criteria of the neighborhood food source is based on their fitness value using the proportionate reproduction scheme (Goldberg & Deb, 1991).

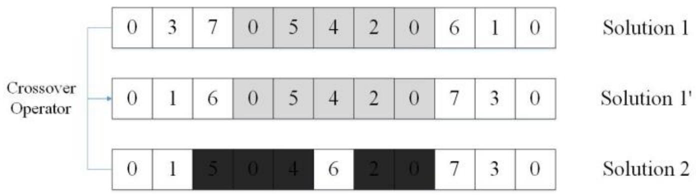


Fig. 5e: Crossover Operator

3.1.4. Onlooker Bee Phase

The information provided by employed bees is shared in the hive with onlooker bees. The onlooker bees select the food sources based on the abundance of nectar. In the MC-ABC algorithm, the selection rule is according to the probability of each food source and is computed by the roulette wheel mechanism using equation (24), which encourages the reproduction of favorable solutions in the iterative procedure. With known probability, certain abundant food sources may be selected by onlooker bees several times. With this mechanism, good solutions are amended by neighborhood operators to speed up the convergence, and it helps establish efficient and intensive searching.

$$p_{li} = \frac{fit(x_{li})}{\sum_{i=1}^{SN} fit(x_{li})}, \forall i \tag{24}$$

3.1.5. Proposed Modified Scout Bee Phase

In the traditional ABC algorithm, the food source may be abandoned by the employed bees on the condition that there is improvement of the fitness value of the corresponding food source after numerous iterations. An employed bee, then, becomes a scout bee to assign another food source randomly, which is constructed the same as the process described in the initialization phase, if there is no update on the objective function from several iterations. In the aforementioned phase of the MC-ABC algorithm, communication among colonies is prohibited to ensure independent searching by the employed bee, and onlooker bee. Even with the same tactic of neighbor searching practices in multiple colony, the solutions from each colony are still diversified. The pseudo code is described in Table 3.

With regard to the views of the probabilistic algorithm, the ABC algorithm does not guarantee the same near optimal solution every time. In order to reduce the heterogeneity of the optimal solution, the MC-ABC algorithm is introduced herein with two solution amendment methods – Elitism-based selection, Random-based selection - that contribute to the scaling down of the solution gap, delivering a better solution. The global exchange tactics of elitism-based selection is to replace the worst solution by the best solution from other colonies, while the approach of random-based selection is to substitute the unfavorable solution from a random solution in a separate colony. The approach of abandoning a solution still takes part in the scout bee phase, when the worst solution and the selected candidate for the replacement are from the same colony. In other multiple colonies strategies, the immigration parameter r is an adaptive information exchange method for the problem of poor and dominated solutions, but this parameter is not suggested in the MC-ABC algorithm. The scout bee performs a negative feedback mechanism to abandon a solution, which is a similar mechanism to the others to avoid excessive domination. Therefore, the control parameter $limit$ is already an adaptive replacement rule. With the purpose of minimizing excessive information exchange among colonies, the maximum tolerance of neighbor searching is defined as $limit = C * SN * Dim$.

The global exchange method is to replace the entire known solution with the worst solution rather than a replacement of a certain sub-route. In the employed bee phase, we apply the crossover operator instead of whole solution substitution to avoid premature convergence, allowing diversity of the whole solution space. Nevertheless, the logic of global information exchange is to transfer knowledge when it is necessary, and is not more frequent than the crossover operator in the employed bee phase. From this point of view, there must be a certain feasibility of exploitation in an intermediate solution by the signal of the control parameter $limit$ with the supreme fitness value. Therefore, the global exchange method is only considered with a tightened criterion for abandoning a solution.

3.1.5.1. Elitism-based Multiple-Colonies ABC algorithm (EBMC-ABC)

The preference of a solution is defined by the duration of a solution without being updated – $trial$ in VRP model. The elite solution is interpreted as an intermediate solution with the least value of $trial$, and, at the same time, the worst solution is defined as a solution which is overwhelmed by the maximum resistance of local searching. As mentioned, there are two possible outcomes of elite selection: an initialized solution or a better intermediate solution. The probability of selecting candidates from an intermediate solution is in large proportion due to the intensive inside-colony searching implemented by employed bees and onlooker bees. Hence, this selection criterion guarantees further exploitation from the best current intermediate solution, and contributes to an overall best solution.

3.1.5.2. Random-based Multiple-Colonies ABC algorithm (RBMC-ABC)

Random-based selection falls into another dimension in generating "best-fit" solutions. The way to define the best fit is too subjective to interpret the appropriateness of an intermediate solution. It is hard to understand the true meaning of trapping in local optima, since the definition of elite in the previous section refers to the intermediate solution, with the near stage of being mature that leads to subjectivity. Thus, a random-based selection approach is a programming scheme to select an intermediate from any iterative stage. Although this approach does not guarantee the best current intermediate solution to be selected, at least the possibility of being trapped in local optima is greatly reduced, and allows the iterative process to move the way from subjective to stochastic selection.

Table 3: The Pseudo Code of Scout Bee Phase in overall Colonies Selection

Enhanced Scout Bee Phase with Elite Selection among All Colonies	**Enhanced Scout Bee Phase with Random Selection among All Colonies**

Integer $BestColony \leftarrow 0; BestFood \leftarrow 0$
Integer $k \leftarrow 0; i \leftarrow 0;$
$limit = C * SN * Dim$
For $k < C$
    For $i < SN$
        If $Trial(x_{ki}) \geq Trial(x_{WorstColony,WorstFood})$
        Then
            $WorstColony = k$
            $WorstFood = i$
        End If
        If $Trial(x_{ki}) \leq Trial(x_{WorstColony,WorstFood})$
        Then
            $BestColony = k$
            $BestFood = i$
        End If
    End For
End For
If $Trial(x_{WorstColony,WorstFood}) \geq limit$
    If $BestColony \neq WorstColony$
    Then
        $x_{WorstColony,WorstFood} \leftarrow x_{BestColony,BestFood}$
        $fit(x_{WorstColony,WorstFood}) \leftarrow fit(x_{BestColony,BestFood})$
        $fun(x_{WorstColony,WorstFood}) \leftarrow fun(x_{BestColony,BestFood})$
        $Trial(x_{WorstColony,WorstFood}) \leftarrow 0$
    Else
        Generate a random feasible solution with the consideration of vehicle
        Calculate the functional value, $fun(x_{WorstColony,WorstFood})$
        Evaluate new solution through the fitness value
        $Trial(x_{WorstColony,WorstFood}) \leftarrow 0$
    End If
End If
$\text{Iteration} \leftarrow \text{Iteration} + 1$
	Integer $RandomColony \leftarrow 0; RandomFood \leftarrow 0$
	Integer $k \leftarrow 0; i \leftarrow 0;$
	$limit = C * SN * Dim$
	For $k < C$
	    For $i < SN$
	        If $Trial(x_{ki}) \geq Trial(x_{WorstColony,WorstFood})$
	        Then
	            $WorstColony = k$
	            $WorstFood = i$
	        End If
	  End For
	End For
	Integer RN
	$\text{RandomColony} \leftarrow \text{RN}, \text{RN} = 1,2, \ldots, \text{C}$
	$\text{RandomFood} \leftarrow \text{RN}, \text{RN} = 1,2, \ldots, \text{SN}$
	If $Trial(x_{WorstColony,WorstFood}) \geq limit$
	    If $RandomColony \neq WorstColony$
	    Then
	        $x_{WorstColony,WorstFood} \leftarrow x_{RandomColony,RandomFood}$
	        $fit(x_{WorstColony,WorstFood}) \leftarrow fit(x_{RandomColony,RandomFood})$
	        $fun(x_{WorstColony,WorstFood}) \leftarrow fun(x_{RandomColony,RandomFood})$
	        $Trial(x_{WorstColony,WorstFood}) \leftarrow 0$
	    Else
	        Generate a random feasible solution with the consideration of vehicle
	        Calculate the functional value, $fun(x_{WorstColony,WorstFood})$
	        Evaluate new solution through the fitness value
	            $Trial(x_{WorstColony,WorstFood}) \leftarrow 0$
	        End If
	$\text{Iteration} \leftarrow \text{Iteration} + 1$

End of Scout Bee Phase with Elite Selection	**End of Scout Bee Phase with Random Selection**

3.2. Rerouting strategies

The local search mechanism is involved in rerouting strategies once the drivers arrive the next node. The exchange of certain remaining customers with the consideration of traffic factor allows a more precious and agile delivery plan to reduce the uncertain of road traffic. The method herein is to extract the latest traffic information once the truck is arrived to the next customers and apply local search to the remaining node for optimization, as shown in Figure 5f. Due to a small solution space in each sub-route, the approaches of local search optimizing the remaining node through the exchange of certain elements to reduce the time-length of the corresponding sub-route is sufficient (Zhang et al., 2014). By incorporating the local search mechanism, the solution quality can be improved and guided by latest traffic in order to achieve a minimized travel time by receding horizon control. The rerouting strategy is applied to sub-route exchange in the same vehicle in order to reduce the impact on other delivering vehicles in our model. The exchange of global elements is not limited to same vehicle or other vehicles. The rule of element exchanges depends on the business model and technological availability of the company. The availability of real time information and location tracking system allow rerouting strategies in fleet management.

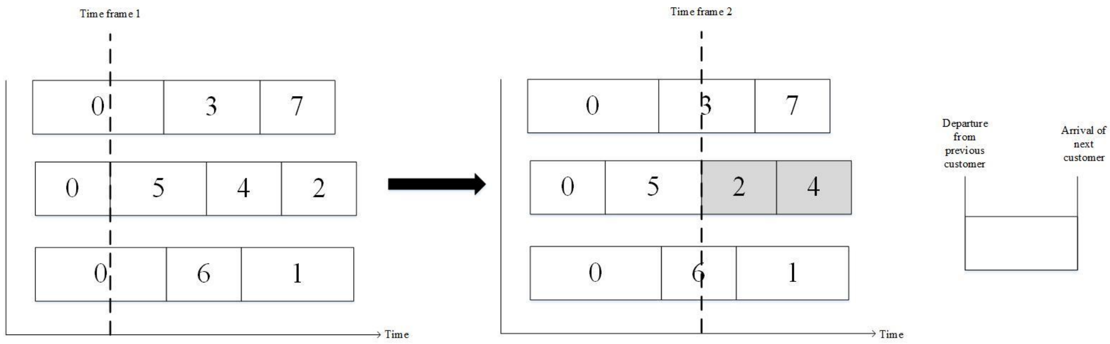


Fig. 5f: Re-routing strategy

4. Numerical Experiment

In this section, the performance of the proposed MC-ABC algorithm is evaluated by using several CVRP instances from the literature in comparison with the benchmark ABC algorithm and modified ABC algorithm. All algorithms were coded in C++ language with visual studio 2013, and all numerical experiments were performed on a computer with Intel Core i7 3.60 GHz CPU and 16.0 GB Ram under Window 7 Enterprise 64-bit operating environment.

The test running is done based on classical CVRP instances called sets A, B, M and P, as proposed by Augerat (1995). The instance data can be retrieved at http://www.coin-or.org/. The MC-ABC algorithm outperforms the OABC and MABC for the above test instances. In the problem formulation, the objective is to minimize the travel time $min\, f = \sum_i \sum_j \sum_k t_{ij} * x_{ijk}$, which is the composition of travel distance, vehicle speed and traffic factor, using the formula of $t_{ij} = \frac{d_{ij}}{sp(1-TDE)}$. For the purpose of performance comparison, the vehicle speed sp is set as 1 and traffic density estimation TDE as 0 to compare with the best-known solutions from the literature. Hence, the total travel distance d is assumed to be identical as the total travel time t. The differences in parameters between ABC and MC-ABC are shown in Table 4. For the parameter setting, the number of employed bees and onlooker bees is equivalent to the half of the colony size, which is 25. Karaboga (2005) mentions that it given an acceptable convergence speed for exploitation and exploration with a bee colony size of 50. Based on the research conducted by Szeto et al, the maximum iteration time $Maxlterations$ is equal to the multiplication of a fixed integer number 2000 and the number of customers n, which is sufficient to converge a near optimal solution ($Maxlterations = 2000n$). Each instance was run 20 times to summarize an average performance. As for the single bee colony algorithm (OABC and MABC), the $limit$ is formulated as the multiplication of food source SN and dimension size Dim. As for the multiple colonies algorithm (EBMC-ABC and RBMC-ABC), the number of colonies C equal to 3, that is sufficient to converge the solution in a single objective model, and the $limit$ is $C * SN * Dim$ to reduce excessive information exchange among all colonies. Other parameters remain unchanged to reduce the parameter tuning. These results are also evaluated together with best-known results from the literature, collected from literature to measure the deviation between exact algorithms and proposed heuristics algorithms (Blecker et al., 2008; Nemhauser & Bienstock, 2005; Toth & Vigo, 2014; Wang, 2013).

Table 4: Parameter Comparison between ABC, Modified ABC and MC-ABC Algorithms

Algorithm	Number of colony	Number of solutions in each colony, SN	Neighbor search operators in employed bee phase	Criterion to abandon a solution, $limit$	Tactics to create new solution
OABC	1	$CS/_2$	Swap, Insert, Reverse Operators	$SN \times Dim$	Random generated
MABC	1	$CS/_2$	Swap, Insert, Reverse, Crossover Operators	$SN \times Dim$	Random generated

EBMC-ABC/ RBMC-ABC	C	$CS/_2$	Swap, Insert, Reverse, Crossover Operators	$SN \times Dim \times \mathrm{C}$	Elitism-based selection, or Random-based selection

4.1. Operator Analysis

With an aim of evaluating the effectiveness of the multiple colonies strategy in the ABC algorithm, the single colony approach (SCA), the multiple colonies elitism-based approach (EB-MCA) and the multiple colonies random-based approach (RB-MCA) are initially evaluated on an instance of B-n78-k10, taking into account only one operator at a time: the swap, insert, reverse and crossover operators with 100,000 iteration. The three operators provide randomness exchange in the solution x_i space to exploit the current solution to find a better neighbor solution $\overline{x_i}$. The performance of each operator is indicated by comparing the maximum value, minimum value, average value, standard deviation and deviation from best-known solution (DFBK) over 20 runs. The best-known objective value of B-n78-k10 is 1221 (Blecker et al., 2008). In accordance with the numerical result, no matter whether the single or multiple colony ABC algorithm, no operators are able to converge to the best-known solution and no significant improvement was found after the 60,000 iterations. According to Figures 6a, 6b and 6c, it can be observed that the MC-ABC algorithm converges rapidly after 10,000 iterations, except the crossover operator in Figure 6d when only a single operator is involved. This is because the knowledge transfer at the initial stage is immature, which leads to a spread of the intermediate solutions. After a certain learning progress of each individual colony, a wise information exchange shows significant value, which contributes to the global bee colonies performance rather than single colony operation. Although the single colony ABC algorithm gradually exploits and explores a better solution, the multiple colonies ABC algorithm yields a more attractive solution after it reaches critical mass, which shows the diversity and enhancement of the exploitation with massive searching.

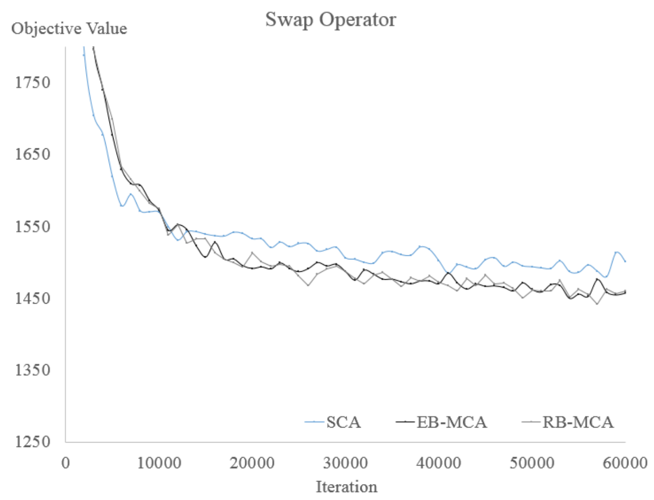


Fig.6a: Comparison of single and multiple colonies with swap operator

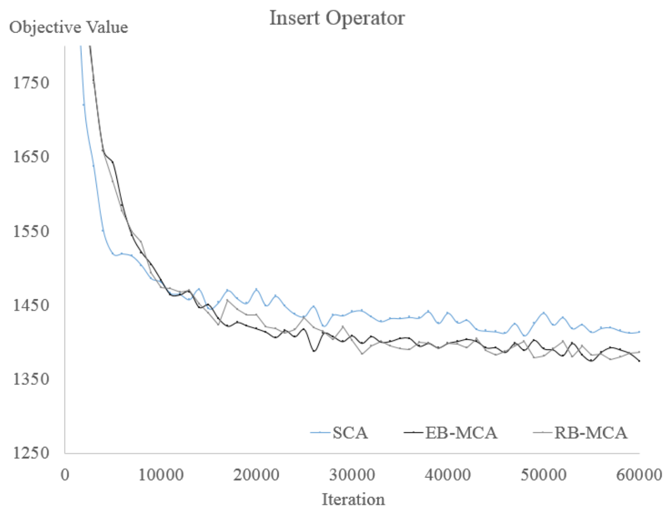


Fig.6b: Comparison of single and multiple colonies with insert operator

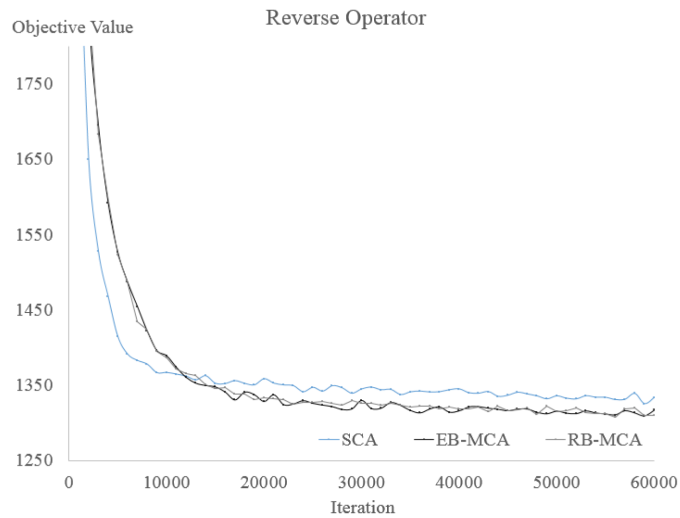


Fig.6c: Comparison of single and multiple colonies with reverse operator

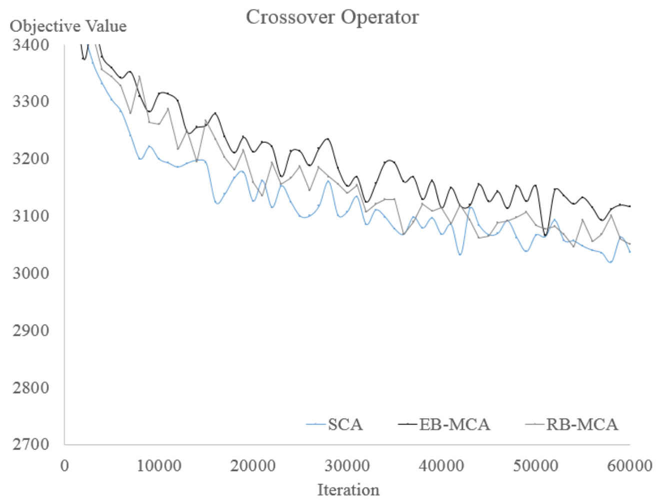


Fig.6d: Comparison of single and multiple colonies with crossover operator

In general, the deviation from best known solution (DFBK) of the above operators' analysis indicates that MC-ABC algorithm provides a better solution than the single colony ABC algorithm. f^* denoted the best known solution from the literature, while f is the approximate solution generated by meta-heuristics. $DFBK$ is an indicator to measure the variation of the results between proposed algorithm and exact method under the same instance. A small value in $DFBK$ denoted a narrow gap between approximate and optimal values.

$$DFBK = \frac{f - f^*}{f^*} \times 100 \qquad (25)$$

In Table 5, the solution quality produced by the MC-ABC algorithm with reverse operator achieves a closer value from the best-known solution. The best known result of each instance is shown in section 4.4. The MC-ABC algorithms with reverse operator surpass the other operators with 7.31% and 7.85% deviation from optimum respectively, while the insert operator in the MC-ABC algorithms have 13.52% and 12.57% DFBK over the original ABC algorithm for the B-n78-k10 instance. A-n80-k10, M-n200-k10 and P-n101-k4 share the similar pattern. In addition, the proposed algorithms are able to develop attractive solutions with the same iteration time for all operators, except the crossover operator in other instances.

Table 5: Experimental results by different operators for the A-n80-k10, B-n78-k10, M-n200-k17 and P-n101-k4 Instances

In	O	p	Swap	Insert	Reverse	Crossover

		ABC	EBMC-ABC	RBMC-ABC	ABC	EBMC-ABC	RBMC-ABC	ABC	EBMC-ABC	RBMC-ABC	ABC	EBMC-ABC	RBMC-ABC
A-n80-k10	Min	1990	2029	2028	2000	1977	1961	1905	1875	1899	3530	3656	3536
	Max	3889	3889	3889	3889	3889	3889	3889	3889	3889	3889	3910	3889
	Avg	2189.85	2161.80	2165.60	2156.35	2120.55	2110.95	2045.70	2016.30	2026.90	3717.10	3789.60	3722.05
	SD	401.89	407.37	406.32	408.93	416.97	419.96	434.30	441.14	438.53	78.56	65.25	85.04
	DFBK	24.21%	**22.62%**	22.84%	22.31%	20.28%	**19.74%**	16.04%	**14.37%**	14.97%	**110.84%**	114.95%	111.12%
B-n78-k10	Min	1396	1425	1354	1362	1329	1346	1310	1286	1286	2846	3002	2876
	Max	1558	1514	1515	1467	1423	1418	1364	1326	1337	3129	3260	3179
	Avg	1500.40	1460.55	1457.20	1414.00	1386.05	1374.50	1333.75	1310.30	1316.80	3036.25	3116.55	3051.25
	SD	39.56	25.22	36.66	26.60	23.18	19.34	12.62	12.88	12.84	69.73	78.15	83.44
	DFBK	22.88%	19.62%	**19.34%**	15.81%	**13.52%**	12.57%	9.23%	**7.31%**	7.85%	**148.67%**	155.25%	149.90%
M-n200-k17	Min	1852	1861	1838	1646	1560	1622	1507	1462	1468	5303	5470	5304
	Max	2030	1941	1979	1825	1741	1729	1591	1541	1544	5637	5740	5705
	Avg	1955.50	1899.05	1917.45	1747.50	1673.65	1670.60	1552.10	1510.55	1501.65	5508.00	5596.60	5544.05
	SD	44.37	22.86	35.92	44.04	41.97	31.99	21.88	20.96	20.62	89.51	84.41	98.78
	DFBK	53.37%	**48.95%**	50.39%	37.06%	31.27%	**31.03%**	21.73%	18.47%	**17.78%**	**332.00%**	338.95%	334.83%
P-n101-k4	Min	897	890	876	780	763	775	721	725	715	2391	2452	2374
	Max	994	976	965	859	829	834	767	742	750	2534	2605	2581
	Avg	957.85	939.65	931.80	824.05	804.15	805.25	747.40	734.50	734.70	2471.55	2525.95	2476.80
	SD	28.49	20.00	25.65	23.71	15.86	16.55	13.24	5.54	7.64	45.11	37.84	49.51
	DFBK	40.65%	37.98%	**36.83%**	21.01%	**18.08%**	18.25%	9.75%	**7.86%**	7.89%	**262.93%**	270.92%	263.70%

4.2. Algorithms Performance Analysis

To evaluate the performance of the MC-ABC algorithms, the original ABC algorithm is selected for benchmarking in algorithm analysis. The ABC algorithm is a well-known algorithm, which has been shown to be competitive with other swarm intelligence-based and population-based algorithms. In this experiment, the original ABC algorithm and modified ABC algorithm are selected for evaluation, as all the selected algorithms share similar natures in exploitation and exploration techniques, by honeybee swarm behavior, and are able to provide a strong support to illustrate the robustness of multiple colonies strategies. The converging processes of all the selected algorithms are demonstrated in Figure 6e and Table 6. The corresponding DFBK of EBMC-ABC and RBMC-ABC algorithms are 3.61% and 4.12%, is less divergence than the original ABC and modified ABC algorithms. A more detail performance analysis for hybrid model is shown in section 4.4.

Table 6: Experimental results by hybrid operators for the B-n78-k10 Instance

Instance	Operator	Hybrid			
		OABC	MABC	EBMC-ABC	RBMC-ABC
B-n78-k10	Min	1299	1256	1246	1225
	Max	1330	1296	1291	1292
	Avg	1314.05	1281.45	1265.05	1271.30
	SD	8.28	10.59	15.43	17.83
	DFBK	7.62%	4.95%	**3.61%**	4.12%

Similar iteration results are given in Figure 6e. The MC-ABC algorithms require a learning process after attaining the critical mass after around 8,000 iterations. The multiple colonies help the ABC algorithm to escape from local optima, and are able to introduce better intermediate solution to other colonies, which help to search more precisely for the global optimum. In addition, the actual population of solutions is multiplied in conducting an extensive searching, and it is possible to exploit better intermediate or optimal solutions, and then share with other colonies. With this collective approach in searching, the overall performance of the MC-ABC algorithm can be improved by enhancing the ability of exploitation and utilizing the exploration capacity.

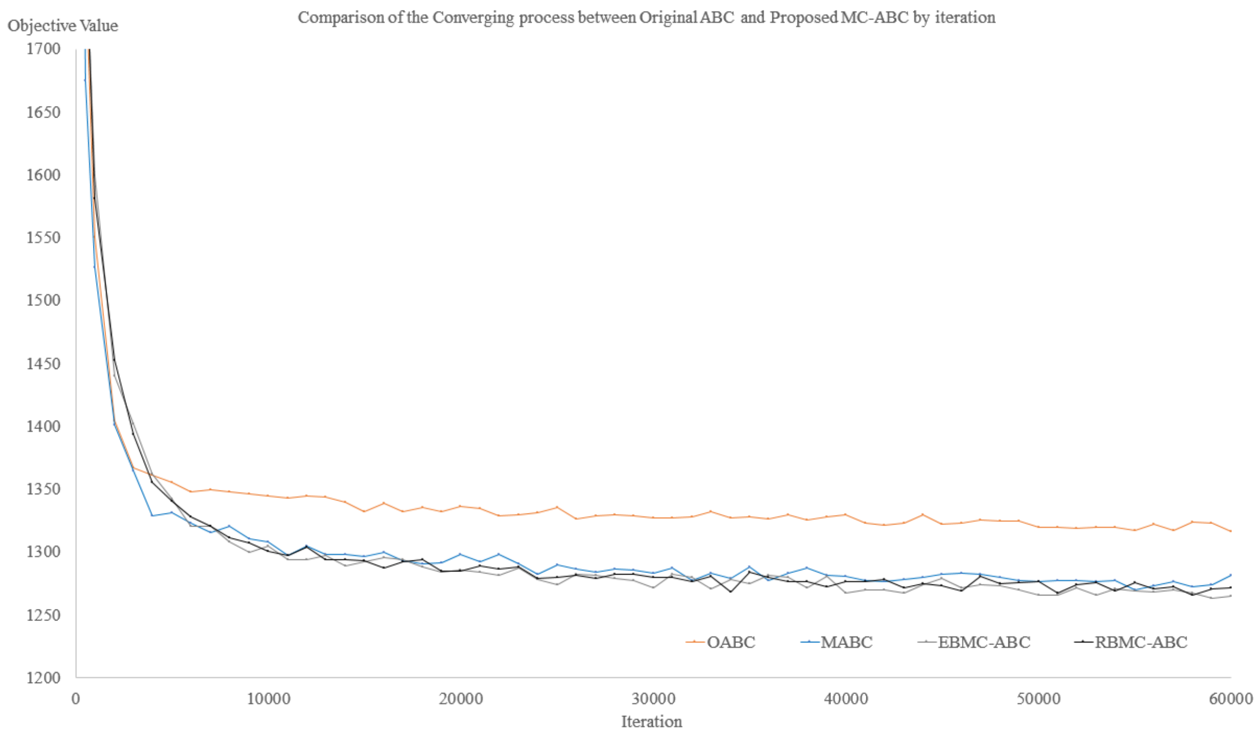


Fig.6e: Comparison between single and multiple colonies with hybrid opreators by iteration

In order to provide a fair judgement on the performance of the proposed algorithm, the computational result based on the same computational time (60 seconds) are given in Figure 6f. Although the algorithm structure of EBMC-ABC and RBMC-ABC algorithms are more complex comparing with OABC and MABC algorithms, the performance of proposed algorithms by given a maximum computational time and maximum iterations are similar. Figure 6f indicates that OABC and MABC algorithms are not able to convergence to the optimal after 40 seconds computational time.

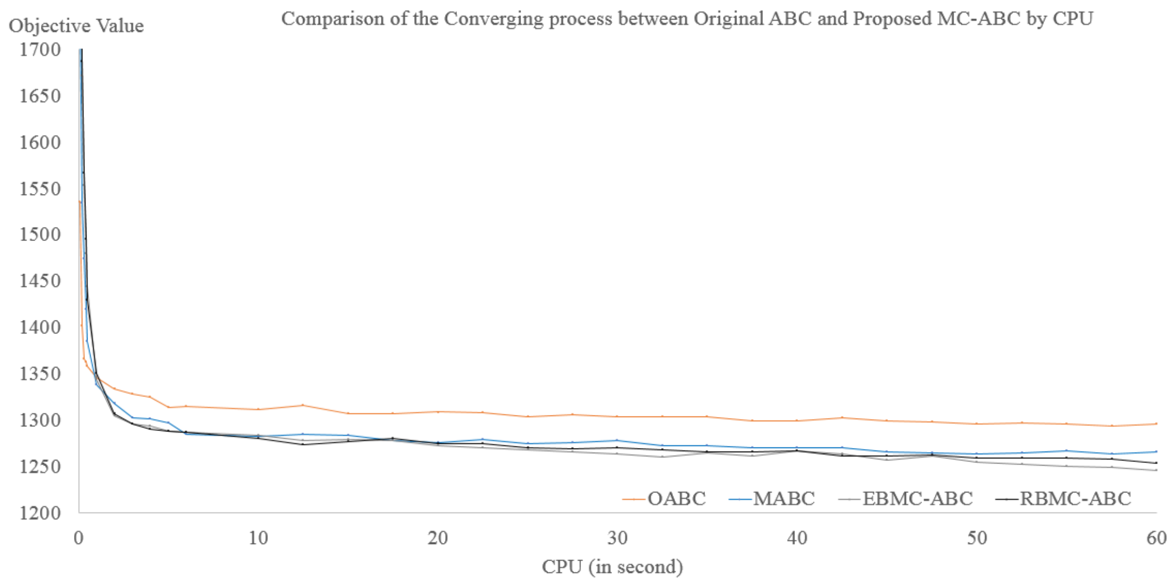


Fig.6f: Comparison between single and multiple colonies with hybrid opreators by CPU

4.3. Computational Results

In this study, four classical VRP instance models (Set A, B, M and P) following Augerat (1995) are adopted to evaluate the proposed algorithms and help summarize the overall performance. The best objective value, and mean of objective value over 20 runtimes are included to illustrate the strength of exploitation and the average performance. The results generated from the original ABC algorithm are treated as a benchmark to study the percentage of improvement $Imp\,\%$ of the proposed algorithms. The highest value of $Imp\,\%$ implies a better average performance than the others. The computation time CPU is measured in minutes. Minimum improvement $Min\,Imp$ and maximum improvement $Max\,Imp$ measure the minimum and maximum percentage gap of the objective values between the selected algorithm and the benchmarked ABC algorithm, while the average improvement $Avg\,Imp$ denotes the gap of the standard performance between two algorithms over 20 runtimes. The average improvements for all VRP class models solved by the EMBC-ABC algorithm are 3.245%, 2.58%, 9.48% and 2.78%. Obviously, EBMC-ABC and RBMC-ABC algorithm are capable of discovering a better solution, especially the EBMC-ABC algorithm. The results in Table 7 indicate that the selection criterion of elite candidates using the control parameter $limit$ was able to extract better intermediate solutions from neighbor bee colonies, share the prosperity and contribute to the entire bee colonies society.

Table 7: The summary of the improvement of experimental result of OABC, MABC, proposed EBMC-ABC and RBMC-ABC algorithms

VRP Class	Modified ABC			Elitism-based MC-ABC			Random-based MC-ABC		
	Min Imp	Avg Imp	Max Imp	Min Imp	Avg Imp	Max Imp	Min Imp	Avg Imp	Max Imp
A-Class	0.07	2.90	5.47	**0.08**	**3.25**	**6.67**	**0.08**	3.12	6.12
B-Class	**0.16**	2.14	5.43	**0.16**	**2.58**	**7.70**	**0.16**	2.45	**7.70**
M-Class	5.98	7.60	9.01	**8.45**	**9.48**	10.27	8.14	9.11	**10.55**
P-Class	**0.00**	2.43	5.80	**0.00**	**2.78**	**6.87**	**0.00**	2.68	6.66

4.4. Exploitation ability analysis

In Tables 8a to 8d, the best solutions found by selected algorithms, within 20 runtimes, are extracted, and compared with the best-known solution BKS to examine the ability of selection algorithm in balancing exploitation and exploration. CPU time in minutes is included in the same table. In Figure 7, the deviation from the best-known solution as a percentage of each selected algorithm, in studying the ability of exploitation and the mean of DFBK from all subclass with 95% confidence level, are measured. The results demonstrate that the MC-ABC algorithms have the ability to escape from local optimal and further exploit the result in moving towards the optimal solution. The confidence interval for the mean of $\mathrm{DFBK}\,\%$ from the MC-ABC algorithms outperform the OABC and MABC, especially in the A, B, and P Class problems, with less than 1% error of the confidence interval. It is shown that the MC-ABC algorithms are able to maintain the balance of exploitation and exploration to have a better determination in searching for the near optimal.

Table 8a: The comparison between the best-known solution and computational result for A class model

Instance	BKS	Original ABC			Modified ABC			Elitism-based MC-ABC (EBMC-			Random-based MC-ABC		
		Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU
A-n32-k5	784 [a]	**784**	**0.00%**	0.05	**784**	**0.00%**	0.20	**784**	**0.00%**	0.54	**784**	**0.00%**	0.56
A-n33-k5	661 [a]	**661**	**0.00%**	0.07	**661**	**0.00%**	0.21	**661**	**0.00%**	0.61	**661**	**0.00%**	0.61
A-n33-k6	742 [a]	**742**	**0.00%**	0.08	**742**	**0.00%**	0.21	**742**	**0.00%**	0.63	**742**	**0.00%**	0.63
A-n34-k5	778 [a]	780	0.26%	0.08	**778**	**0.00%**	0.22	**778**	**0.00%**	0.65	**778**	**0.00%**	0.65
A-n36-k5	799 [a]	811	1.50%	0.09	**799**	**0.00%**	0.25	**799**	**0.00%**	0.72	**799**	**0.00%**	0.71
A-n37-k5	669 [a]	**669**	**0.00%**	0.10	**669**	**0.00%**	0.26	**669**	**0.00%**	0.73	**669**	**0.00%**	0.73
A-n37-k6	949 [a]	955	0.63%	0.10	**949**	**0.00%**	0.26	**949**	**0.00%**	0.72	**949**	**0.00%**	0.71
A-n38-k5	730 [a]	**730**	**0.00%**	0.10	**730**	**0.00%**	0.26	**730**	**0.00%**	0.75	**730**	**0.00%**	0.74
A-n39-k5	822 [a]	828	0.73%	0.11	**822**	**0.00%**	0.28	**822**	**0.00%**	0.78	**822**	**0.00%**	0.78
A-n39-k6	831 [a]	833	0.24%	0.11	**831**	**0.00%**	0.28	**831**	**0.00%**	0.79	**831**	**0.00%**	0.79
A-n44-k6	937 [a]	958	2.24%	0.13	**937**	**0.00%**	0.34	**937**	**0.00%**	0.96	**937**	**0.00%**	0.95
A-n45-k6	944 [a]	966	2.33%	0.14	948	0.42%	0.36	949	0.53%	0.99	**944**	**0.00%**	0.98
A-n45-k7	1146 [a]	1175	2.53%	0.14	**1146**	**0.00%**	0.36	**1146**	**0.00%**	1.00	**1146**	**0.00%**	0.99
A-n46-k7	914 [a]	925	1.20%	0.14	**914**	**0.00%**	0.38	**914**	**0.00%**	1.04	**914**	**0.00%**	1.05

A-n48-k7	1073 [a]	1095	2.05%	0.15	**1073**	**0.00%**	0.37	**1073**	**0.00%**	1.13	**107**	**0.00%**	1.11
A-n53-k7	1010 [a]	1026	1.58%	0.18	1015	0.50%	0.38	**1010**	**0.00%**	1.36	**101**	**0.00%**	1.34
A-n54-k7	1167 [a]	1191	2.06%	0.18	1172	0.43%	0.38	**1167**	**0.00%**	1.36	**1167**	**0.00%**	1.36
A-n55-k9	1073 [a]	1088	1.40%	0.19	**1073**	**0.00%**	0.40	**1073**	**0.00%**	1.41	**1073**	**0.00%**	1.41
A-n60-k9	1354 [a]	1406	3.84%	0.22	1358	0.30%	0.45	1355	0.07%	1.66	**1354**	**0.00%**	1.63
A-n61-k9	1034 [a]	1078	4.26%	0.23	**1035**	**0.10%**	0.41	**1035**	**0.10%**	1.69	**1035**	**0.10%**	1.67
A-n62-k8	1288 [a]	1360	5.59%	0.23	1299	0.85%	0.44	1300	0.93%	1.82	**1292**	**0.31%**	1.84
A-n63-k9	1616 [a]	1650	2.10%	0.24	1636	1.24%	0.44	**1627**	**0.68%**	1.87	**1627**	**0.68%**	1.84
A-n63-	1314 [a]	1375	4.64%	0.24	1320	0.46%	0.44	**1319**	**0.38%**	1.83	1320	0.46%	1.83
A-n64-k9	1401 [a]	1473	5.14%	0.25	1425	1.71%	0.46	**1412**	**0.79%**	1.91	1416	1.07%	1.89
A-n65-k9	1174 [a]	1221	4.00%	0.25	1181	0.60%	0.46	1178	0.34%	1.87	**1174**	**0.00%**	1.87
A-n69-k9	1159 [a]	1196	3.19%	0.27	**1166**	**0.60%**	0.51	**1166**	**0.60%**	2.04	1170	0.95%	2.02
A-n80-	1763 [a]	1896	7.54%	0.30	1810	2.67%	0.68	**1774**	**0.62%**	2.74	1786	1.30%	2.76

[a] Obtained from the literature (Wang, 2013)

Table 8b: The comparison between the best-known solution and computational result for B class model

Instance	BKS	Original ABC			Modified ABC			Elitism-based MC-ABC (EBMC-			Random-based MC-ABC		
		Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU
B-n31-k5	672 [b]	**672**	**0.00%**	0.06	**672**	**0.00%**	0.16	**672**	**0.00%**	0.55	**672**	**0.00%**	0.55
B-n34-k5	788 [b]	**788**	**0.00%**	0.08	**788**	**0.00%**	0.19	**788**	**0.00%**	0.63	**788**	**0.00%**	0.63
B-n35-k5	955 [b]	**955**	**0.00%**	0.09	**955**	**0.00%**	0.20	**955**	**0.00%**	0.68	**955**	**0.00%**	0.68
B-n38-k6	805 [b]	807	0.25%	0.10	**805**	**0.00%**	0.23	**805**	**0.00%**	0.79	**805**	**0.00%**	0.79
B-n39-k5	549 [b]	550	0.18%	0.10	**549**	**0.00%**	0.24	**549**	**0.00%**	0.79	**549**	**0.00%**	0.79
B-n41-k6	829 [b]	836	0.84%	0.12	**829**	**0.00%**	0.26	**829**	**0.00%**	0.83	**829**	**0.00%**	0.84
B-n43-k6	742 [b]	746	0.54%	0.12	**742**	**0.00%**	0.29	**742**	**0.00%**	0.91	**742**	**0.00%**	0.91
B-n44-k7	909 [b]	924	1.65%	0.13	**909**	**0.00%**	0.29	**909**	**0.00%**	0.95	**909**	**0.00%**	0.93
B-n45-k5	751 [b]	**751**	**0.00%**	0.13	**751**	**0.00%**	0.31	**751**	**0.00%**	1.00	**751**	**0.00%**	1.01
B-n45-k6	678 [b]	699	3.10%	0.13	680	0.29%	0.30	**678**	**0.00%**	0.99	**678**	**0.00%**	0.98
B-n50-k7	741 [b]	742	0.13%	0.17	**741**	**0.00%**	0.38	**741**	**0.00%**	1.21	**741**	**0.00%**	1.22
B-n50-k8	1312 [b]	1328	1.22%	0.16	1316	0.30%	0.36	**1313**	**0.08%**	1.17	1316	0.30%	1.18
B-n51-k7	1032 [b]	**1032**	**0.00%**	0.17	**1032**	**0.00%**	0.38	**1032**	**0.00%**	1.22	**1032**	**0.00%**	1.23
B-n52-k7	747 [b]	752	0.67%	0.17	**747**	**0.00%**	0.40	**747**	**0.00%**	1.29	**747**	**0.00%**	1.29
B-n56-k7	707 [b]	718	1.56%	0.19	**707**	**0.00%**	0.38	**707**	**0.00%**	1.46	**707**	**0.00%**	1.45
B-n57-k7	1153 [b]	**1153**	**0.00%**	0.20	**1153**	**0.00%**	0.33	**1153**	**0.00%**	1.47	**1153**	**0.00%**	1.49
B-n57-k9	1598 [b]	1637	2.44%	0.20	1602	0.25%	0.33	**1600**	**0.13%**	1.49	1604	0.38%	1.49
B-n63-k10	1496 [b]	1569	4.88%	0.25	1508	0.80%	0.41	1507	0.74%	1.81	**1497**	**0.07%**	1.81
B-n64-k9	861 [b]	896	4.07%	0.25	868	0.81%	0.39	862	0.12%	1.89	**861**	**0.00%**	1.86
B-n66-k9	1316 [b]	1355	2.96%	0.26	1320	0.30%	0.37	**1319**	**0.23%**	2.03	1322	0.46%	2.03
B-n67-k10	1032 [b]	1077	4.36%	0.27	**1036**	**0.39%**	0.38	1037	0.48%	2.05	1037	0.48%	2.04
B-n68-k9	1272 [b]	1315	3.38%	0.27	1289	1.34%	0.39	1279	0.55%	2.06	**1277**	**0.39%**	2.05
B-n78-k10	1221 [b]	1299	6.39%	0.35	1241	1.64%	0.51	**1236**	**1.23%**	2.57	1239	1.47%	2.55

[b] Obtained from the literature (Giaglis et al., 2004)

Table 8c: The comparison between the best-known solution and computational result for M class model

Instance	BKS	Original ABC			Modified ABC			Elitism-based MC-ABC (EBMC-			Random-based MC-ABC		
		Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU
M-n101-k10	820 [c]	877	6.95%	0.53	837	2.07%	1.38	827	0.85%	4.34	**824**	**0.49%**	4.33
M-n121-k7	1034 [d]	1244	20.31%	0.73	1127	8.99%	2.20	**1078**	**4.26%**	6.66	1082	4.64%	6.70
M-n151-k12	1015 [d]	1142	12.51%	1.15	1054	3.84%	2.36	**1048**	**3.25%**	9.44	1056	4.04%	9.56
M-n200-k16	1274 [d]	1477	15.93%	1.91	1363	6.99%	3.17	1335	4.79%	10.03	**1333**	**4.63%**	10.03
M-n200-k17	1275 [d]	1489	16.78%	1.43	1369	7.37%	4.64	1344	5.41%	11.56	**1328**	**4.16%**	11.61

[c] Obtained from the literature (Nemhauser & Bienstock, 2005)

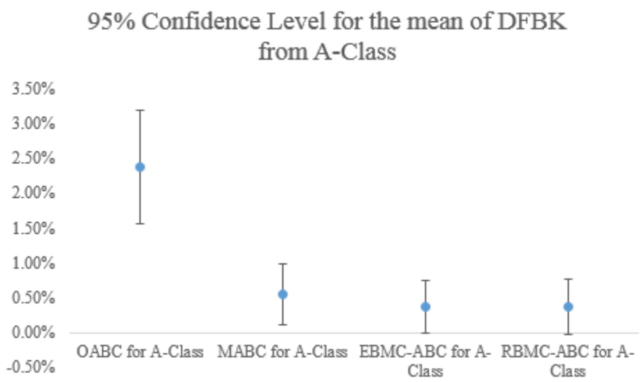
[d] Obtained from the literature (Toth & Vigo, 2014)

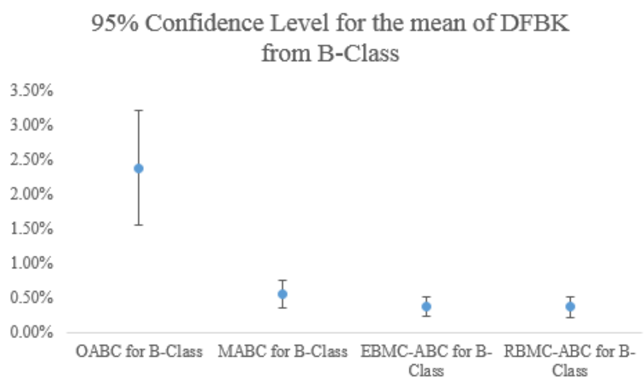
Table 8d: The comparison between the best-known solution and computational result for P class model

Instance	BKS	Original ABC			Modified ABC			Elitism-based MC-ABC			Random-based MC-ABC		
		Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU	Best	DFBK %	CPU
P-n16-k8	450 [e]	**450**	**0.00%**	0.02	**450**	**0.00%**	0.06	**450**	**0.00%**	0.22	**450**	**0.00%**	0.21
P-n19-k2	212 [e]	**212**	**0.00%**	0.03	**212**	**0.00%**	0.07	**212**	**0.00%**	0.25	**212**	**0.00%**	0.25
P-n20-k2	216 [e]	**216**	**0.00%**	0.03	**216**	**0.00%**	0.08	**216**	**0.00%**	0.27	**216**	**0.00%**	0.27
P-n21-k2	211 [e]	**211**	**0.00%**	0.04	**211**	**0.00%**	0.09	**211**	**0.00%**	0.28	**211**	**0.00%**	0.29
P-n22-k2	216 [e]	**216**	**0.00%**	0.04	**216**	**0.00%**	0.09	**216**	**0.00%**	0.31	**216**	**0.00%**	0.32
P-n22-k8	603 [e]	**603**	**0.00%**	0.05	**603**	**0.00%**	0.10	**603**	**0.00%**	0.33	**603**	**0.00%**	0.33
P-n23-k8	529 [e]	**529**	**0.00%**	0.05	**529**	**0.00%**	0.11	**529**	**0.00%**	0.36	**529**	**0.00%**	0.35
P-n40-k5	458 [e]	**458**	**0.00%**	0.11	**458**	**0.00%**	0.26	**458**	**0.00%**	0.82	**458**	**0.00%**	0.82
P-n45-k5	510 [e]	516	1.18%	0.13	**510**	**0.00%**	0.31	**510**	**0.00%**	0.90	**510**	**0.00%**	0.87
P-n50-k7	554 [e]	565	1.99%	0.16	**554**	**0.00%**	0.37	**554**	**0.00%**	0.88	**554**	**0.00%**	0.87
P-n50-k8	631 [e]	633	0.32%	0.16	633	0.32%	0.36	**631**	**0.00%**	0.88	**631**	**0.00%**	0.87
P-n50-k10	696 [e]	713	2.44%	0.17	702	0.86%	0.37	700	0.57%	0.89	**698**	**0.29%**	0.88
P-n51-k10	741 [e]	768	3.64%	0.17	**741**	**0.00%**	0.39	**741**	**0.00%**	0.93	742	0.13%	0.92
P-n55-k7	568 [e]	587	3.35%	0.19	570	0.35%	0.44	**568**	**0.00%**	1.05	**568**	**0.00%**	1.06
P-n55-k8	576 [e]	588	2.08%	0.19	**576**	**0.00%**	0.44	**576**	**0.00%**	1.05	577	0.17%	1.05
P-n55-k10	694 [e]	708	2.02%	0.20	**694**	**0.00%**	0.43	695	0.14%	1.04	699	0.72%	1.03
P-n60-k10	744 [e]	773	3.90%	0.22	746	0.27%	0.37	**744**	**0.00%**	1.09	745	0.13%	1.09
P-n60-k15	968 [e]	1002	3.51%	0.24	971	0.31%	0.38	972	0.41%	1.23	**968**	**0.00%**	1.21
P-n65-k10	792 [e]	828	4.55%	0.25	801	1.14%	0.43	**792**	**0.00%**	1.37	**792**	**0.00%**	1.37
P-n70-k10	827 [f]	875	5.80%	0.29	**831**	**0.48%**	0.43	836	1.09%	1.57	840	1.57%	1.54
P-n76-k4	593 [e]	625	5.40%	0.31	596	0.51%	0.59	595	0.34%	2.10	**593**	**0.00%**	2.08
P-n76-k5	627 [e]	662	5.58%	0.31	630	0.48%	0.56	630	0.48%	1.89	**628**	**0.16%**	1.93
P-n101-k4	681 [e]	721	5.87%	0.50	691	1.47%	1.14	**684**	**0.44%**	3.80	685	0.59%	3.82

[e] Obtained from the literature (Blecker et al., 2008)

[f] Obtained from the literature (Toth & Vigo, 2014)





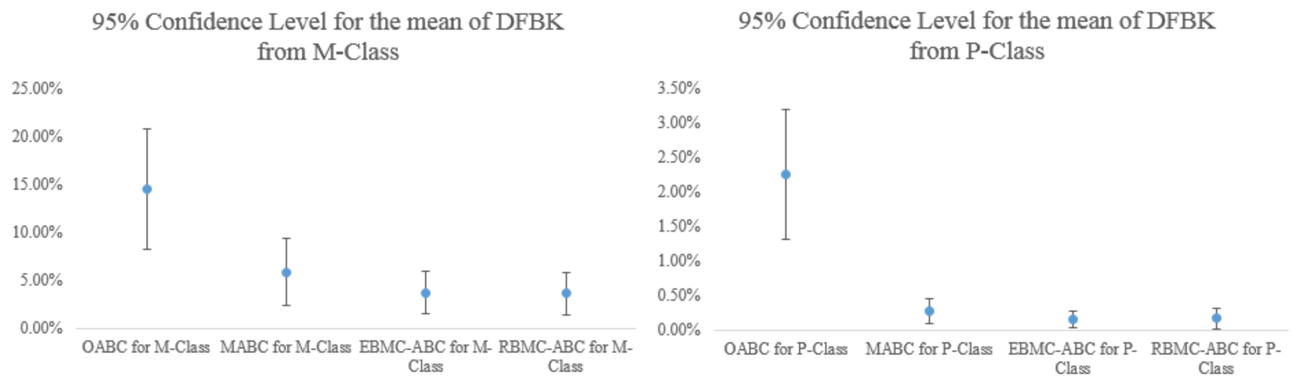


Fig.7: The confidence level for the mean of DFBK from A, B, M, P class model

5. Numerical study on OVRP with Traffic Factors with a Case Study

The methodology is discussed to generate the CVRP solution in offline performance in the test instances, for benchmarking purposes. In this section, traffic factors are examined with a real case scenario. Logistics companies always encounter the OVRP when the optimal delivery schedule is needed. A logistics service provider in Hong Kong, which has 128 clients, is employed as a case study. Most of the clients are located at high traffic network spots. In the current vehicle scheduling process of the company, the routes are constructed manually at regular periods. The company management noticed that customer loyalty might be easily affected by the risk of late delivery due to traffic congestion. A particular vehicle fleets with 35 customers is selected to demonstrate the OVRP model in Figure 8. The vehicle speed sp under free-flow condition is defined as 50 km/hr.

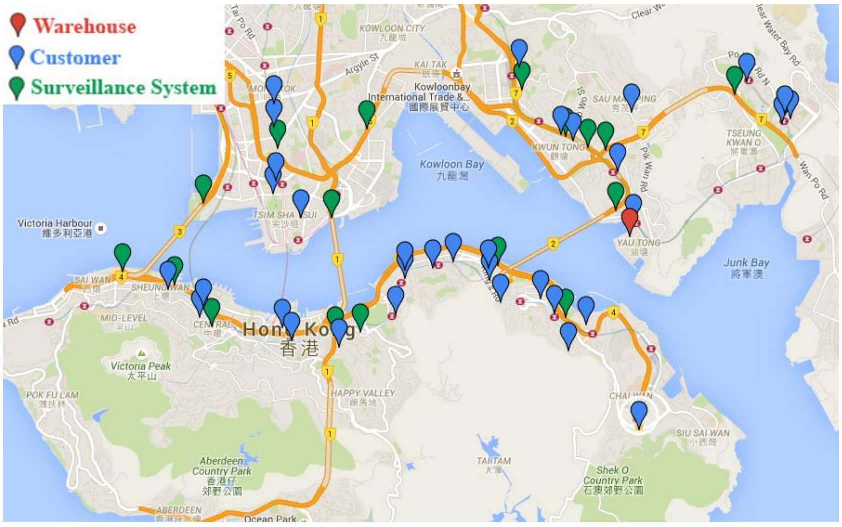


Fig. 8: Location of warehouse, customer and surveillance system in case study

5.1. Traffic Density Estimation

The traffic density fluctuates, and depends on the current traffic flow and the number of vehicles. Figure 9 shows an example of traffic density estimation in the Hung Hom Cross Harbor Tunnel, between Hong Kong Island and Kowloon. Figure 10 shows the demo application of OVRP system and provide real time solution for local search. The real time arrival and processing information are upload to the server for real time operation with mobile technology. The system evaluates the traffic density TDE of the corresponding traffic images, and this information provides a reference point to the OVRP model to measure the current traffic condition and travel time. Each sub-route is selected in order to evaluate the sub-route traveling time with the help of a local search. If there is a significant traffic jam in a pre-planned sub-route, the local search operators tries to find a possible solution by swapping the sub-route sequences (Zhang et al., 2014). In our model, a global exchange method in real time system is prohibited, as the carried items on a specific truck are homogeneous to the other truck. The business nature of the case company is to provide equipment hygienic service. Each truck carries the corresponding items to serve the customers before the start of delivery. Therefore, it is not feasible to process global exchanges, but local search is allowed.

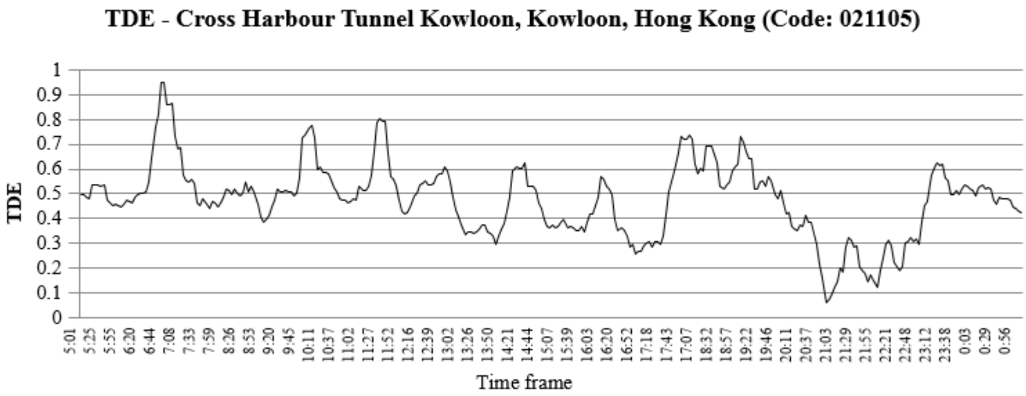


Fig. 9: Traffic density estimation on 24rd December, 2014

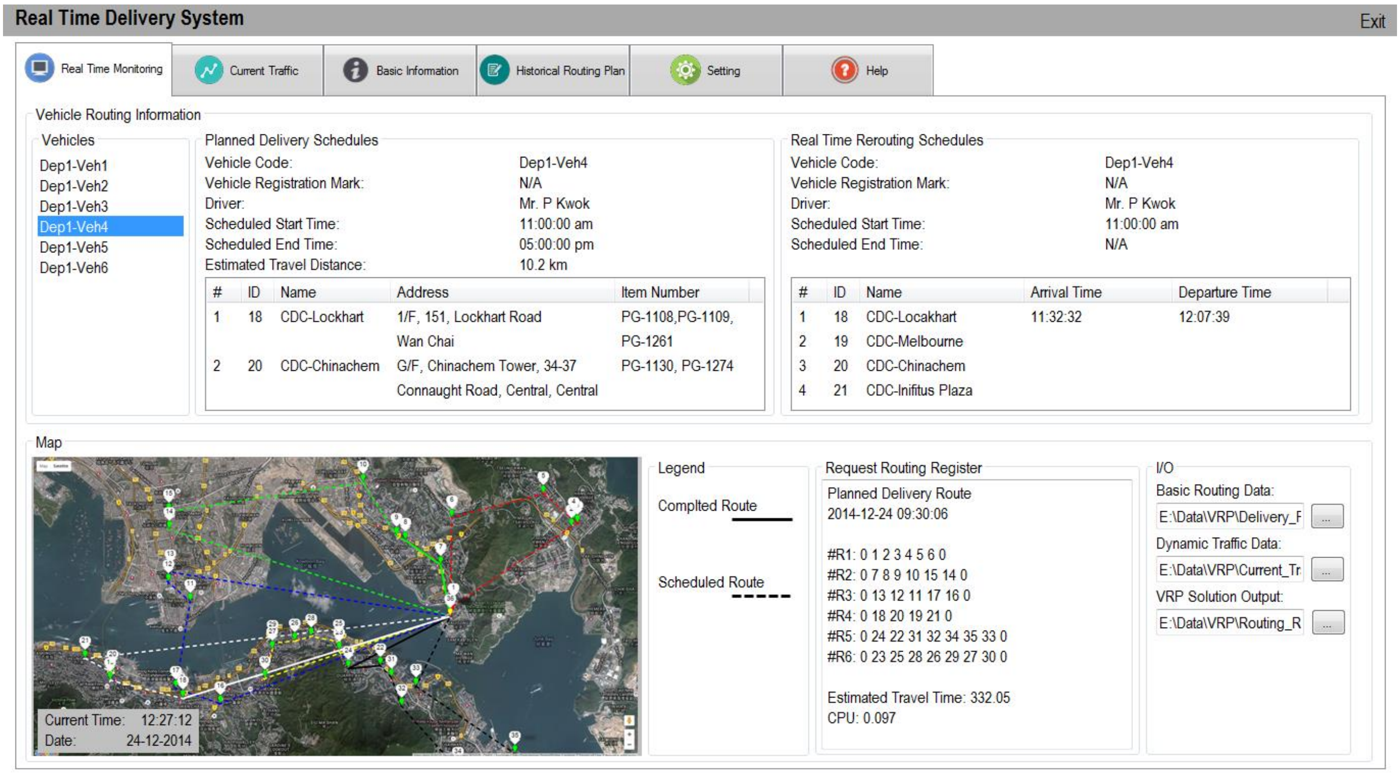


Fig. 10: The real time experimental environments for OVRP

The pre-planned solution and simulated routing performance from 11am to 5pm on 24rd December, 2014 are shown in Table 9. The estimated delivery time without traffic consideration is 332.05 minutes, while the simulated delivery time are usually larger than 640 minutes. This shows a very large difference in travel time between the pre-planning and real-time schedules. The dynamic behavior of the traffic status is uncontrollable, however, OVRP maintains a better performance with traffic environmental changes.

Table 9: Online performance of real time vehicle routing result

Time frame	Pre-planned solution						Estimated Time
Free-flow	Route 1: [0,1,2,3,4,5,6,0]; Route 2:[0,7,8,9,10,15,14,0]; Route 3: [0,13,12,11,17,16,0]; Route 4: [0,18,20,19,21,0]; Route 5: [0,24,22,31,32,34,35,33,0]; Route 6: [23,25,28,26,29,27,30,0]						332.05 minutes
Date	24th Dec	25th Dec	26th Dec	27th Dec	28th Dec	29th Dec	30th Dec

Total travel time estimated by initial static traffic	738 mins	790 mins	797 mins	803 mins	875 mins	879 mins	867 mins
Total travel time with periodic rerouting strategy	674 mins	657 mins	664 mins	635 mins	646 mins	677 mins	744 mins
Percentage of improvement	-8.67%	-16.84%	-16.69%	-20.92%	-26.17%	-22.98%	-14.19%

6. Conclusions and future work

In this research, the approach for traffic density estimation is considered and is processed to meet the real time needs for retrieving historical and current traffic conditions in various applications. A focus is put on the real time approach between traffic density estimation and modern vehicle routing problems. Referring to the dynamic changes of the road traffic information, the routing schedules can be re-planned and re-scheduled to reduce the impact of logistics risk. In addition, two MC-ABC algorithms are presented for solving the re-optimization process of OVRP under time-dependent traffic congestion. The experimental results indicate that MC-ABC algorithms outperform the original ABC and modified ABC, and achieve better results, particularly the EBMC-ABC. The proposed algorithms are evaluated on a set of well-known CVRP instances in term of total travel distance, has and is compared with benchmarked ABC and modified ABC algorithms to show the ability of exploitation and exploration in the searching space. The proposed MC-ABC algorithms have exploitation ability after the iteration reaches to the critical mass by selecting the best-known intermediate solution. The EBMC-ABC algorithm generally converges well in exploitation, while maintaining solution diversity. Future research is recommended in the following aspects. (1) Obtain more accurate vehicle scheduling using information on anticipated and unanticipated events, like weather predict and road construction. (2) Extend a similar approach to study a global sub-route exchange method to retain flexibility in agile delivery.

Reference

Ahmadi-Javid, A., & Seddighi, A. H. (2013). A location-routing problem with disruption risk. *Transportation Research Part E: Logistics and Transportation Review, 53*, 63-82.

Ai, T. J., & Kachitvichyanukui, V. (2009). A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research, 36*(5), 1693-1702.

Arnott, R., De Palma, A., & Lindsey, R. (1991). Does providing information to drivers reduce traffic congestion? *Transportation Research Part A: General, 25*(5), 309-318.

Augerat, P. (1995). *Approche polyèdrale du problème de tournées de véhicules.* Institut National Polytechnique de Grenoble-INPG.

Baker, B. M., & Ayechew, M. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research, 30*(5), 787-800.

Bell, J. E., & McMullen, P. R. (2004). Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics, 18*(1), 41-48.

Bhagade, A. S., & Puranik, P. V. (2012). Artificial bee colony (ABC) algorithm for vehicle routing optimization problem. *Int. J. Soft Comput. Eng, 2*, 329-333.

Blecker, T., Kersten, W., & Gertz, C. (2008). *Management in Logistics Networks and Nodes: Concepts, Technology and Applications* (Vol. 8): Erich Schmidt Verlag GmbH & Co KG.

Chen, H.-K., Hsueh, C.-F., & Chang, M.-S. (2006). The real-time time-dependent vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review, 42*(5), 383-408.

Chen, X., Xi, C., & Cao, J. (2015). Research on moving object detection based on improved mixture Gaussian model. *Optik-International Journal for Light and Electron Optics, 126*(20), 2256-2259.

Du, T., Wang, F., & Lu, P.-Y. (2007). A real-time vehicle-dispatching system for consolidating milk runs. *Transportation Research Part E: Logistics and Transportation Review, 43*(5), 565-577.

Ehmke, J. F., Campbell, A. M., & Urban, T. L. (2015). Ensuring service levels in routing problems with time windows and stochastic travel times. *European Journal of Operational Research, 240*(2), 539-550.

Errico, F., Desaulniers, G., Gendreau, M., Rei, W., & Rousseau, L.-M. (2016). A priori optimization with recourse for the vehicle routing problem with hard time windows and stochastic service times. *European Journal of Operational Research, 249*(1), 55-66.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management science, 40*(10), 1276-1290.

Gendreau, M., Laporte, G., & Séguin, R. (1995). An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation Science, 29*(2), 143-155.

Gendreau, M., Laporte, G., & Séguin, R. (1996). A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations research, 44*(3), 469-477.

Giaglis, G. M., Minis, I., Tatarakis, A., & Zeimpekis, V. (2004). Minimizing logistics risk through real-time vehicle routing and mobile technologies: Research to date and future trends. *International Journal of Physical Distribution & Logistics Management, 34*(9), 749-764.

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms, 1*, 69-93.

Haughton, M. A. (2002). Route reoptimization's impact on delivery efficiency. *Transportation Research Part E: Logistics and Transportation Review, 38*(1), 53-63.

Haughton, M. A. (2007). Assigning delivery routes to drivers under variable customer demands. *Transportation Research Part E: Logistics and Transportation Review, 43*(2), 157-172.

Huang, S.-H., & Lin, P.-C. (2010). A modified ant colony optimization algorithm for multi-item inventory routing problems with demand uncertainty. *Transportation Research Part E: Logistics and Transportation Review, 46*(5), 598-611.

Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization*. Retrieved from

Kim, S., Lewis, M. E., & White, C. C. (2005). Optimal vehicle routing with real-time traffic information. *IEEE Transactions on Intelligent Transportation Systems, 6*(2), 178-188.

Lee, D.-S. (2005). Effective Gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(5), 827-832.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2009a). A Lagrangian heuristic for the real-time vehicle rescheduling problem. *Transportation Research Part E: Logistics and Transportation Review, 45*(3), 419-433.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2009b). Real-time vehicle rerouting problems with time windows. *European Journal of Operational Research, 194*(3), 711-727.

Marinakis, Y. (2015). An improved particle swarm optimization algorithm for the capacitated location routing problem and for the location routing problem with stochastic demands. *Applied soft computing, 37*, 680-701.

Mavrovouniotis, M., & Yang, S. (2012). *Ant colony optimization with memory-based immigrants for the dynamic vehicle routing problem.* Paper presented at the 2012 IEEE Congress on Evolutionary Computation.

Mavrovouniotis, M., & Yang, S. (2014). *Elitism-based immigrants for ant colony optimization in dynamic environments: Adapting the replacement rate.* Paper presented at the 2014 IEEE Congress on Evolutionary Computation (CEC).

Mavrovouniotis, M., & Yang, S. (2015). Ant algorithms with immigrants schemes for the dynamic vehicle routing problem. *Information Sciences, 294*, 456-477.

Mezura-Montes, E., & Velez-Koeppel, R. E. (2010). *Elitist artificial bee colony for constrained real-parameter optimization.* Paper presented at the IEEE congress on evolutionary computation.

Nemhauser, G., & Bienstock, D. (2005). *Integer programming and combinatorial optimization*. Berlin: Springer.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research, 225*(1), 1-11.

Ritzinger, U., Puchinger, J., & Hartl, R. F. (2016). A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research, 54*(1), 215-231.

Sen, R., Cross, A., Vashistha, A., Padmanabhan, V. N., Cutrell, E., & Thies, W. (2013). *Accurate speed and density measurement for road traffic in India.* Paper presented at the Proceedings of the 3rd ACM Symposium on Computing for Development.

Szeto, W., Wu, Y., & Ho, S. C. (2011). An artificial bee colony algorithm for the capacitated vehicle routing problem. *European Journal of Operational Research, 215*(1), 126-135.

Toklu, N. E., Gambardella, L. M., & Montemanni, R. (2014). A multiple ant colony system for a vehicle routing problem with time windows and uncertain travel times. *Journal of Traffic and Logistics Engineering Vol, 2*(1).

Toth, P., & Vigo, D. (2014). *Vehicle routing: problems, methods, and applications* (Vol. 18): Siam.

Wang, J. (2013). *Management innovations for intelligent supply chains*. Hershy, PA: Business Science Reference.

Waters, C. (1989). Vehicle-scheduling problems with uncertainty and omitted customers. *Journal of the Operational Research Society, 40*(12), 1099-1108.

Wilson, M. C. (2007). The impact of transportation disruptions on supply chain performance. *Transportation Research Part E: Logistics and Transportation Review, 43*(4), 295-320.

Xiang, Y., & Zhou, Y. (2015). A dynamic multi-colony artificial bee colony algorithm for multi-objective optimization. *Applied soft computing, 35*, 766-785.

Xiang, Y., Zhou, Y., & Liu, H. (2015). An elitism based multi-objective artificial bee colony algorithm. *European Journal of Operational Research, 245*(1), 168-193.

Yan, S., Lin, J.-R., & Lai, C.-W. (2013). The planning and real-time adjustment of courier routing and scheduling under stochastic travel times and demands. *Transportation Research Part E: Logistics and Transportation Review, 53*, 34-48.

Yang, S. (2008). Genetic algorithms with memory-and elitism-based immigrants in dynamic environments. *Evolutionary Computation, 16*(3), 385-416.

Zhang, S., Lee, C., Choy, K., Ho, W., & Ip, W. (2014). Design and development of a hybrid artificial bee colony algorithm for the environmental vehicle routing problem. *Transportation Research Part D: Transport and Environment, 31*, 85-99.

Zhang, T., Chaovalitwongse, W. A., & Zhang, Y. (2012). Scatter search for the stochastic travel-time vehicle routing problem with simultaneous pick-ups and deliveries. *Computers & Operations Research, 39*(10), 2277-2290.