 doi:10.20944/preprints202201.0130.v1

Article

Metaheuristic Algorithms Based on Compromise Programming for Multi-Objective Urban Shipment Problem

Tung Son Ngo [1,2,*], Jafreezal Jaafar [1], Izzatdin Abdul Aziz [1], Muhammad Umar Aftab [3], Hoang Giang Nguyen [2] and Ngoc Anh Bui [2]

[1] Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, 32610 Seri Iskandar, Perak Darul Ridzuan, Malaysia; jafreez@utp.edu.my, izzatdin@utp.edu.my.
[2] Information and Communication Department, FPT University,10000 Hanoi, Vietnam; giangnhhe150094@fpt.edu.vn, anhbn5@fe.edu.vn.
[3] Department of Computer Science, National University of Computer and Emerging Sciences, Chiniot-Faisalabad Campus, Chiniot, 35400, Pakistan; ms.umaraftab@yahoo.com.
* Correspondence: sonnt69@fe.edu.vn

Abstract: The Vehicle Routing Problem (VRP) and its variants are found in many fields, especially logistics. In this study, we introduced an adaptive method to a complex VRP. It combines multi-objective optimization and several forms of VRPs with practical requirements for an urban shipment system. The optimizer needs to consider terrain and traffic conditions. The proposed model also considers customers' expectations, shipper considerations as goals, and the common goal like transportation cost. We offered compromise programming to approach the multi-objective problem by decomposing the original multi-objective problem into a minimized distance-based problem. We designed a hybrid version of the Genetic algorithm with the Local Search algorithm to solve the proposed problem. We evaluate the effectiveness of the proposed algorithm with the Tabu Search algorithm, the original Genetic algorithm on the tested dataset. The results show that our method is an effective decision-making tool for the multi-objective VRP and an effective solver for the new variation of VRP.

Keywords: Multi Objective Optimization; VRP; Compromise Programming; Genetic Algorithm, Local Search, Tabu Search; Metaheuristics; Combinatorial Optimization.

1. Introduction

1.1. Vehicle Routing Problem and Variants

In a logistics system, transportation plays an essential role in moving materials from supplier to manufacturer, processing plants to the next step in the production process or transporting finished products to customers. This scheduling and planning process needs to be calculated before the actual operation. However, this is not an easy task because many resources such as machines and vehicles need to be arranged. The problem is called VRP [**1**]. A capable fleet of vehicles must serve a dispersed group of customers geographically at a minimal cost. We can express the VRP with visitations of the vehicles to customers through a graph as $G = (V, E)$ where $V = \{v_0, v_1, \dots, v_n\}$ be the set of nodes. $v_1, \dots, v_n$ represent the customers to be visited from the depot v_0. E is an edge set interlinking two locations where $E = \{(i,j) | (i,j) = 0,1, \dots, n, i \neq j\}$. Fundamental decisions are made in the VRP regarding customer assignment to vehicles and the sequence of customers assigned to each vehicle [**2**]. Many studies on VRP have been conducted, and accordingly, many variants of VRP have also been identified. Some of the more widely known variants include:

- Capacitated Vehicle Routing Problem (C-VRP): refers to the limitation of vehicle capacity for classical VRPs. The system uses multiple vehicles and the total demand of each route does not exceed the vehicle capacity [**3**] [**4**]. An extension of C-VRP is when

 doi:10.20944/preprints202201.0130.v1

the vehicles are heterogeneous (CH-VRP), where each vehicle may have a different capacity [**5**], [**6**]. The package of product with different types can be considered as multi product -VRP (MP-VRP) [**7**].

- Multi-Depot Vehicle Routing Problem (MD-VRP): For the classic VRP problem, the path of all vehicles can only start from a warehouse. In MD-VRP, the vehicle departed from multiple warehouses [**7**], . Torres et al. reviewed several variants of the MD-VRP, where it can combine with several constraints such as time windows, batch delivery, heterogeneous fleets, scheduled delivery [**8**].
- Vehicle Routing Problem with Time Window (VRP-TW): is often encountered in many industrial applications. The time window is divided into soft time windows (Delivery not within the period can be a penalty) and hard time windows (Delivery within the period is mandatory). VRP-TW has received much attention from researchers in recent times [**9**], [**10**].
- Multi-trip Vehicle Routing Problem (MT-VRP): in MT-VRP, each vehicle is explicitly allowed to perform multiple trips during its service time in such a way that the total demand of customers served in each route does not exceed its capacity within a given deadline [**11**], [**9**], [**12**].

VRP is a complex problem that challenges many researchers. Different variants or business conditions may require the solution search space to be significantly expanded. This paper introduces a method to solve a new VRP that combines multi-objective optimization (MOP) and different form of VRPs. The proposed solver automatically generates the routings for shippers to deliver packages to urban customers. The urban delivery systems have several characteristics that differ from many ideal environments in terrain, traffic, and order-warehouse conditions. The scheduler allows considering the concern of the business, customer satisfaction, and the employees in the decision-making process. We use MOP-VRP as a shortened name to call the problem.

1.2. Related Works

VRP is a problem with many variations. One can say that each business model can potentially become a new variant with countless goals and constraints based on the success factors of the business. **Table 1** presents some recent studies in this field. In building the optimization model, we can see in these studies that the authors are often only interested in optimizing transportation costs (distance, fuel consumption, transportation cost). The objective functions are usually linear. However, many factors affect this calculation, especially for the case study of transportation in the inner city, such as traffic conditions over time (peak hours) and road conditions (one/two-way roads). In the survey [**18**], the author pointed out that the path optimization problem would raise the trend of real-time path optimization problems considering time-varying factors such as real-time terrain and real-time traffic conditions involved in MOP-VRP. Therefore, a simple representation such as previous studies' use is challenging to cover in many real-life situations. Several studies related to time windows have made customer satisfaction one of the essential goals. However, besides benefiting businesses through cost optimization or satisfying customers, a collaborative economic model requires sharing among stakeholders. Usually, urban transport models often use part-time shippers. If there is not enough attention to their needs, the job will become less attractive, and it won't be easy to create a long-term working environment. The lack of committed employees in a volatile business such as logistics brings more difficult constraints to optimize costs and improve service quality. Therefore, these factors need to be considered in the modeling process to become the objectives to be achieved in the scheduler.

In the decision-making process, decision-makers may not be an individual but a group. Therefore, their considerations are critical to any strategies. In real-world business environments, the optimizer needs to pay more attention to multiple decision criteria to meet customers' requirements. No single solution exists that simultaneously optimizes all objectives of the non-trivial multi-objective scheduling problems. The MOP-VRP is not an

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

exception. Several approaches are available to the MOP [**19**] (including MOP-VRP [**20**], [**21**]). Generally, they are classified into (1) the preferred approach, where decision-makers have the higher information to select the final solution among several options. (2) The non-preference approach assumes that no-decision maker is available, or the decision-maker does not have the higher information to indicate the preferences for the objectives. A user needs only one solution from a practical standpoint, no matter whether the optimization problem is. In the case of MOP, the user is now in a dilemma. The idea MOP procedure always requires finding the trade-off between solutions with a wide range of values for objectives and then choosing one of the obtained solutions using higher-level information. Significantly, the objective space is higher-dimensional. It isn't easy to visualize the solutions to the users [**22**]. In this situation, an adaptive approach to the MOP is a challenge to the researchers. The result of solving a real-world problem usually is an approximation set A of the objective vectors (any element of A does not dominate or is not equal to any other accurate vector in A) and not the Pareto optimal front. Okabe et al. reviewed several metrics for accessing the performance of MOP algorithms [**23**], including cardinality metrics, accuracy metrics, diversity metrics. Regardless of which approach is used, assessing the quality of the solution is one of the issues that need to be carefully considered.

Table 1: Research and corresponding objective functions

Research	Objective Functions	VRP Types	Highlights	Drawbacks
Zhen et al., 2020[**9**]	Minimize traveling time of all the vehicles.	MD-VRP, MT-VRP, VRP-TW	The proposed mixed integer linear programming can describe clearly the business. The proposed metaheuristics can provide optimal solutions on a large scale.	The model is not based on a realistic problem where the data is also randomly selected from the benchmark. The model does not involve several factor such as traffic conditions
Babaee Tirkolaee et al., 2019[**10**]	Minimize the sum of vehicles cost, traveling cost, penalty cost of soft time window.	MT-VRP, VRP-TW	A case study is investigated to evaluate the applicability of the proposed model in the real world. Many business conditions have been considered.	The business rules assume that – The time and cost of a route is the same for all vehicles. This is may not guaranteed in other real-life application. The designed solver can solve the problem in small and medium sizes that only offers near-optimal solutions compared to the CPLEX solver.
Alemany et al., 2018[**7**]	Minimizing distribution cost and distance-base cost.	C-VRP, MP-VRP, MD-VRP	The model was developed from a realistic case study from an oil provider company.	Experiments are conducted on the small-scale dataset. Evaluations of the proposed method have not shown its performance with different techniques.
Pan et al., 2021[**13**]	Minimizing the traveling cost.	MT-VRP, VRP-TW	The routing solver is designed for a vending cafe company to replenish stocks for their geographically dispersed outlets. The proposed method can work on large-scale instances.	Authors simulate the experimented data by randomly creating data based on an existing dataset.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

Ma et al., 2017[**14**]	Minimizing traveling cost.	MD-VRP, VRP-TW	An improved ACO algorithm with some ideal to improve the search speed, is introduced to solve the proposed problem.	The system considers only single depot which is not guaranteed in several applications. The research did not show the evaluations on the proposed algorithm to the existing methods.
Zhang et al., 2020[**15**]	Minimize carbon emission.	MD-VRP	The research develops a new extension model of MD-VRP. The proposed algorithms can deal with large-scale datasets.	The proposed mathematical model and the heuristic algorithm provide better quality than the heuristic but more computational cost.
Nucamendi-Guillén et al., 2021[**16**]	Minimize the cost of transport and contracts.	CH-VRP, MD-VRP	The proposed model was obtained from the real-world business.	Business rules are simple. The designed metaheuristic was only tested with small scale dataset.
Li et al., 2020[**12**]	Minimize completion time of vehicles.	MT-VRP, VRP-TW	The solver can apply to some real-life problem instances. The proposed heuristic algorithm shows a better result to the CPLEX solver.	The model is quite simply cannot adapt to other businesses. The designer did not consider the concerns of different stakeholders in the system.
Shelbourne et al., 2017[**17**]	Minimize the sum of total distance cost and total weighted tardiness.	VRP-TW	Proposed solvers based on heuristic evaluated the performance on several datasets.	The optimization model was based on several assumption that may not be applied to other situations

Usually, researchers often model VRP problems using binary decision variables to represent relationships between vehicles and destinations, vehicles and depots, etc. These links are expanded exponentially to the number of inputs and may not exist solver with a deterministic polynomial time. Therefore, the VRPs are classified in NP-Hard and combinatorial optimization. The state-of-the-art is identified as two main streams of resolution techniques, including exact methods to approximate solution methods (heuristic and meta-heuristic) in the literature. Whereas exact methods provide the optimal that is the best solutions. Exact algorithm includes branch-and-X (bound, cut) [**24**], dynamic programming [**25**], Lagrangian relaxation-based methods [**26**]. (Meta)heuristics include simulated annealing, population-based methods such as evolutionary algorithm [**27**], generally yield near-optimal solutions. The exact method is more suitable for the small size of problems, due to the. However, the logistic system is increasingly facing the larger scale with the increasing number of orders, customers, and vehicles… this is where (meta)heuristics is a better choice due to flexible search capabilities and easy integration to exploit the good properties of different methods.

Many researchers designed heuristic, metaheuristic, and combinations to the VRP and its variant. Samuel Nucamendi-Guillén (2021) [**16**] developed a metaheuristic procedure to find a solution by improving the initial solution using local search algorithms. Zhiwei Liu (2017) [**28**] proposed a method that combines Tabu with mem-brane computing to find the solution for VRPTW. Babaee Tirkolaee E (2019)[**10**] developed simulated annealing (SA), a local search algorithm that can escape the local optimum for the MT-VRPTW in urban waste collection. To solve the MT time-dependent VRP TW. Binbin Pan [**13**] designed a hybrid metaheuristic algorithm, using variable neighborhood descend

 doi:10.20944/preprints202201.0130.v1

(VND) for intensive exploitation and adaptive extensive neighborhood search (ALNS) to direct the inquiry when VND is stuck in a local optimum.

Among several branches of metaheuristics, Evolutionary Algorithms (EA) attract many researchers. For instance, Hari Kurnia (2018) [**29**], Cortes (2018) [**30**], and R Fitriana (2019) [**31**] designed a classical genetic algorithm for the VRP, CVRP, MDVRP, respectively. As for VRP with more features, attributes that reflect the complexity of the real problem, a hybrid genetic algorithm, which improves the solution by implementing a local search heuristic in the initial phase of the genetic, is proposed by Rabbouch (2019) [**32**]. Jalel Euchi (2021)[**33**] solves another complex VRP with drones by a modified hybrid genetic algorithm combined with nearest neighbor heuristic and modifying saving heuristic in the initial phase. While the nearest neighbor heuristic help improve the initial solutions, the saving heuristic keeps the genetic algorithm from the early local optimum. Yanfang Ma (2017) [**14**] proposed an improved ant colony optimization (ACO) combined with the nearest neighbor search method for the MD VRP TW. Wei-heng Zhang (2020) [**15**] designed a 2-stage ACO for MDGVRP, assigning customers to the depot to generate routes. EAs and their hybrid versions have proven effective for single objective VRPs. They can obtain a set of solutions present in a solution process, provide the ability to be easily determined with different types of variables, and do not require any assumptions that make convexity and separability distinction between objectives and related constraints. In general, these algorithms provide a design direction. The suggested search operations are based on different designer arguments. In general, many factors determine this problem, but in our opinion, building a suitable data structure plays a vital role. A good data structure can support stochastic operations and help improve population diversity while ensuring the algorithm's convergence. Multi-Objective EA (MOEA) extends EAs to deal with multi-objective optimization problems. They can be classified based on different features. A widely accepted classification for MOEA considers Pareto-dominance-based, Decomposition-based, and Indicator-based algorithms [**34**]. MOEAs have been applied in several applications [**35**,**36**] that can search for a set of optimal solutions on the Pareto Front. However, it involves much higher-level information, often non-technical, qualitative, and experience-driven, to indicate the final solution with a prohibitive computational cost. This cost is not suitable in many experimental conditions. An efficient approach for MOP-VRP that does not require pre-determine the trade-off between objectives and is applicable to integrate with algorithm design to maintain the solution quality with a reasonable cost has always been a challenge in this field [**37**].

1.3. Contributions

This study presents an adaptive method for a variant of MOP-VRP as a scheduler in the urban delivery system. The model is built around the real-life requirement fit for an urban delivery system. The optimizer needs to provide the solution to archive multiple business conditions that comply with essential factors such as terrain and traffic conditions besides other constraints for VRP. We use compromise programming to approach the proposed MOP. It allows decision-makers to obtain an optimal solution without defining preferences on each objective function in advance. However, if they do, alternative decision strategies are still used generally through the definition of weights to assign the effect of objective functions via the distance function. We designed and compared Tabu search (TA), genetic algorithm (GA), a combination between GA and local search algorithm (HGA) to solve the proposed model on the tested dataset. Our study suggests a new variant of VRP. It can benefit researchers and engineers to develop a better optimizer for variants of VRP. This research also contributes to the developed methodology for Multi objectives scheduling and planning problems [**38**]. The rest of this paper is organized as follows. The proposed model and algorithm are respectively described in Sections 2 and 3. To evaluate the proposed approach, we display the experiments and discussion in Section 4. Finally, section 5 offers a conclusion.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

2. Proposed Model

This study builds a multi-objective optimization solver for the urban shipment problem. This section describes the mathematical optimization model and the approach to the proposed multi-objective optimization problem. The goals to be achieved by the developed scheduler have been thoroughly discussed with logistic managers as the decision-makers. Some important business rules are defined as follows:

- The system is set up with many delivery points corresponding to the customers.
- Packages that need to be delivered have different weights and delivery periods.
- Packages are shipped from various warehouses.
- Each package needs to be collected from the allocated warehouse.
- The delivery time between any 2 locations is time and terrain-dependent. It is estimated based on statistical data from previous shipments.
- Shippers also use vehicles with different payloads and transportation costs.
- To deliver the order, the shipper may need to take several trips.

2.1. Mathematical Formulation

The decision variable represents the whole detailed plan to shippers. Besides them, several dependent variables that are computed from the decision variables are also introduced in the model as follows:

- C represents the set of deliver points/ customers.
- K denotes the set of shippers.
- D stands for the set of warehouses.
- $N = D \cup C$ denotes the set of locations, where the first $|D|$ elements are the locations of the depots, and last $|C|$ elements represent the locations of customers.
- $B \in \mathbb{R}^{|K|}$ is the vector that represent the capacity of shippers, where $B_k \in \mathbb{N}^*$ is the capacity of shipper k^{th}.
- $P \in \mathbb{R}^{|K|}$ is the vector that represent the freight rates of the shippers, where $P_k \in \mathrm{R}^*$ is the freight rate of shipper k^{th}.
- $W \in \mathbb{R}^{|C|}$ is the vector used to illustrate the weight of the orders by customers, where $W_c \in \mathbb{N}^*$ is the weight need to delivery to deliver point c^{th}.
- $L \in \mathbb{R}^{|C|}$ is the vector used to illustrate the load time of the orders by customers, where $L_c \in \mathbb{N}^*$ is the time need to load the package of customer c^{th}.
- $A = \{A^c | A^c \in \mathbb{R}^2, c \in C\}$ is the vector that represent the appointment time of the customer, where $[A^c_{\text{soon}}, A^c_{\text{late}}]$ is respectively describe the appointment time and the time window that customer c^{th} demand for his/her order.
- $M = \{M^k | M^k \in \mathbb{R}^{|N|\times|N|}, k \in K\}$, $M^k = \{M^k_{i,j} | M^k_{i,j} \in \mathbb{N}^*, i, j = 1 \dots |N|\}$ where $M^k_{i,j}$ is the distance if shipper k go from location i^{th} to location j^{th}.
- $T = \{T^k | T^k \in \mathbb{R}^{|N|\times|N|}, k \in K\}$, $T^k = \{T^k_{i,j} | T^k_{i,j} \in \mathbb{N}^*, i, j = 1 \dots |N|\}$ denotes the time consumptions of transportations between locations for shipper k, where $T^k_{i,j}$ represents the time that shipper k^{th} take to travel from location i^{th} to location j^{th}. $T^k_{i,j}$ is computed based on $time(i, j, k, t)$ that is the function to query travelling time of the shipper ks^{th} from location i^{th} to location j^{th}, t stands for start time. The returned value is depended on the traffic condition at time t.
- $U \in \mathbb{R}^{|C|\times|D|}$ is a matrix to represent the warehouse's links that storing the orders and their deliver points. $U_{c,d} = 1$ means the order of the customer c^{th} is keep by the warehouses d^{th}.
- $V = \{V_k | V_k \in \mathbb{N}^*, k \in K\}$ is the vector that represents the number of tours each shipper takes, where V_k is the number of tours of shipper k^{th}.
- The decision variable $O = \{O^k | O^k \in \mathbb{R}^{V_k\times|N|}, k \in K\}$ represents the planned paths for shippers, where $V_k \in \mathbb{N}$ denotes the number of tour that made by the shipper k^{th}

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

to deliver all his/her assigned orders. **Figure 1Error! Reference source not found.** illustrates an example of a planned path.

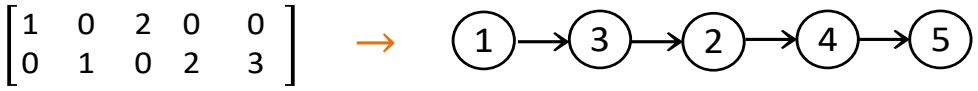

Figure 1. an example of a planned path with $N = 5, V_k = 2$

- $Z = \{Z_{k,v,i,j} | Z_{k,v,i,j} = equal(\{\min(O^k_{v,j}, 1) - \min(O^k_{v,i}, 1), 1\}), \forall\, k = 1 \dots |K|, i = 1 \dots |N|, j = 1 \dots |N|, v = 1 \dots V_k \ \wedge Z_{k,v,S^k_{v,0}} = 1\}$ represents the sequence of visited nodes of the shippers. Shipper k^{th} go to node i^{th} right after node j^{th} in trip v^{th} if $Z_{k,v,i,j} = 1$. 0 otherwise.
- Denotes $S = \{S_c | S_c \in \mathbb{R}^*, c \in C\}$ where S_c is estimated time that shipper deliver to the customer c. If $S_c < A^c_{soon}$ then $S_c = A^c_{soon}$, and the duration $A^c_{soon} - S_c$ is considered as waiting time.
- $Y = \{Y_k | Y_k \in \mathbb{R}^*, k = 1 \dots K\}$ where $Y_k = \sum_{v=1}^{V_k} \sum_{i=1}^{|N|} \sum_{j=1}^{|N|} Z_{k,v,i,j} * T^k_{i,j}$ represents the total traveling time of the shipper k^{th}.

To meet business requirements, the solver must satisfy the following objectives:

- Minimize of transportation cost of all shippers based on types of vehicles. Traveling on long routes increases costs:

$$min\left(f_1(O) = \sum_{k=1}^{|K|} \sum_{v=1}^{V_k} \sum_{i=1}^{|N|} \sum_{j=1}^{|N|} Z_{k,v,i,j} * P_k * M^k_{j,i}\right)$$

- The urban delivery requires punctuality, although it is not a hard constraint to the model—however, the less late delivery, the more satisfied the customer. The optimizer needs to minimize late delivery to the customers:

$$min\left(f_2(O) = \sum_{c=1}^{|C|} late(S_c)\right)$$

Where: $late(S_c) = \begin{cases} 0 & if\ A^c_{late} \geq S_c \\ S_c - A^c_{late} & if\ S_c > A^c_{late} \end{cases}$

- Serving customers is beneficial for businesses. However, it can be traded off by the convenience of the delivery staff. The workforce is usually part-time. Thus, a route that saves shippers waiting time provides a competitive environment. Minimize the waiting time of the shippers is a goal that needs to archive:

$$min\left(f_3(O) = \sum_{c=1}^{|C|} wait(S_c)\right)$$

Where: $wait(S_c) = \begin{cases} 0 & if\ A^c_{soon} \leq S_c \\ A^c_{soon} - S_c & if\ S_c < A^c_{soon} \end{cases}$

- Minimize differences in traveling time of the shippers. The shipper's working time is only counted as travel time. It does not include waiting time. Therefore, this time allocation helps to balance the workload of the shippers.

$$min\left(f_4(O) = \sum_{k=1}^{|K|} \left(\left|Y_k - \frac{1}{|K|} \sum_{i=1}^{|K|} Y_i\right|\right)\right)$$

Subject to:

- All of orders must be delivered:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

$$\sum_{k=1}^{|K|}\sum_{i=|D|+1}^{|N|}\sum_{v=1}^{V_k}\min\left(O_{v,i}^{k},1\right)=|C|$$

- Each deliver point is assigned to only one shipper:

$$\sum_{k=1}^{|K|}\sum_{v=1}^{V_k}\min\left(O_{v,i}^{k},1\right)=1\quad\forall i=|D|+1\ldots|N|$$

- Respecting the capacity of shipper on every trip:

$$\sum_{c=1}^{|C|}min\left(O_{v,c}^{k},1\right)*W_c\le B_k\quad\forall k=1\ldots K,v=1\ldots V_k$$

- Shipper must load the customer's package before delivering to the customer in the same trip.

$$O_{v,c}^{k}-O_{v,d}^{k}\ge U_{c,d}\quad\forall k=1\ldots|K|,v=1\ldots V_k,c=|D|+1\ldots|N|,d=1\ldots|D|$$

- Shipper cannot visit more than one location at the same time.

$$O_{v,i}^{k}\neq O_{v,j}^{k}\ \forall k=1\ldots K,v=1\ldots V_k,i=1\ldots|N|,j=1\ldots|N|,i\neq j,O_{v,i}^{k}\neq 0\ ,O_{v,j}^{k}\neq 0$$

2.2. Compromise Programming for MOP-VRP

Compromise programming (CP) [**39**] is based on the idea not to use any preference information or rely on assumptions about the importance of objectives. The method does not try to find multiple Pareto optimal solutions. Instead, the distance between some reference point and the feasible objective region is minimized to find a single optimal solution, as shown in **Figure 2**. For this purpose, the weighted L_p metrics measure the distance of any solution from the reference point. The ideal objective vector is often used as the reference point:

$$min\left(\sum_{i=1}^{N}w_i\,|f_i(x)-\ z_i^*|^p\right)^{1/p}\quad s.t.\quad x\in X$$

Where x is the decision variable and X is feasible set, $z_i^*=\min_{x\in X}f_i(x)$, p can take any value between 1 and ∞ (in practice normally $p=2$), weight vector $w=\{w_i|w_i\in\mathbb{R}^+\ i=1\ldots N\}$, and N is the number of objective functions. The literature suggests that to normalize the dimensional values in range $[0,1]$ of the distance function. We can rewrite the objective function in form of norm 2 as: $min\left(\sum_{i=1}^{N}w_i\,|\frac{F_i-z_i^*}{z_i^{worst}-z_i^*}|^2\right)^{1/2}$ where $z_i^{worst}=\max_{x\in X}f_i(x)$.

There are many studies that have used CP to approach the MOP problem such as for university timetabling [**40**], [**41**], [**42**], or in team selection [**43**], in knowledge based recommender [**44**], project task assignment [**45**]. However, it may require pre-defined minimal and maximal values of the objective functions. Some of these values are predictable [**43**], most of other cases require to solve the problem as single objective function multiple times, which may be costly. Other studies [**44**] [**46**] show that the referential point may be selected from business estimations could provide better performance for the agents in searching process. However, in this study, the normalization method using both of z^* and z^{worst} brings a better quality of the solution.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

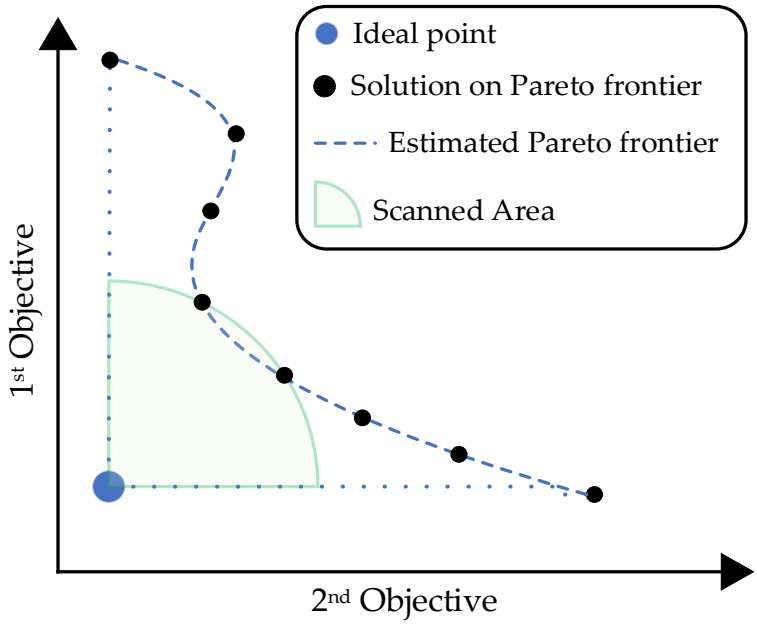

Figure 2: Scanned area of the search process in CP-based approach

3. Proposed Algorithms

In this section, we introduce the proposed algorithms. The main focused algorithm is HGA, an algorithm that combines GA and Local Search. The first part of this section describes GA. We use the same principle for the search agent's stochastic process and data structure for proposed algorithms. The algorithms described after GA share several common strategies. The second part describes how we implement HGA. Another algorithm that does not belong to the class of Evolutionary algorithm is proposed, the TA.

3.1. Genetic Algorithm

GA is one of the most well-known metaheuristic algorithms used to solve NP-hard problems and belong to Evolution Algorithms family [**47**]. The process of natural evolution inspires the idea of GA. The algorithm begins with a random population, with each individual represents a solution for the problem. The final solution is obtained through the evolution of the population. The designed scheme of GA is shown in **Figure 3**. The fundamental difference between our design and traditional flow is that we introduce the repair step to fix the instances violating the constraints during the random process. To initialize the first population, we randomly create the $route$ for each solution of the initial population. With an initial $route$, we get the list of assigned customers for each and then sort them by their demand time in ascending order. After modifying the $route$, the $trip$ is created using the idea that shippers only need to return to the depot whether their job is finished, or the trip's capacity is overloaded. They need to return to the depot.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

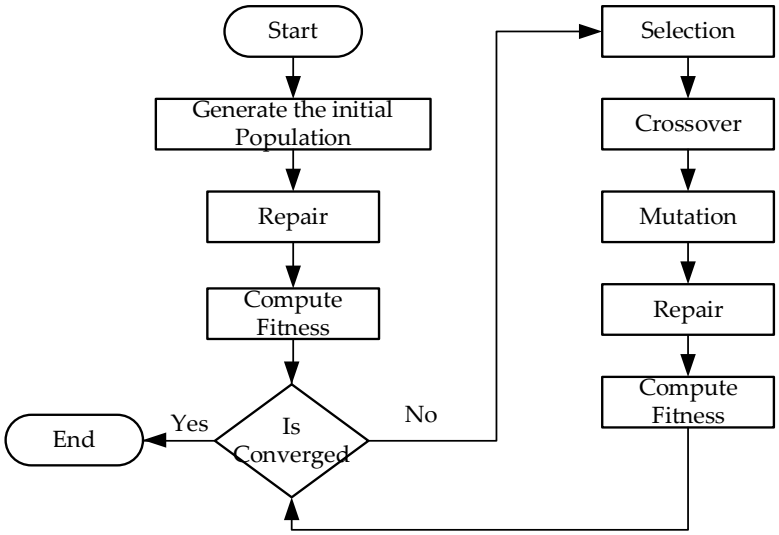

Figure 3: The flow of proposed GA scheme.

1. Initialize the population: The structure of the individual is equivalent to decision variable O as described in the proposed optimization model. We generate the population P is the set of π individuals. For programming convenience, the chromosome is represented by 2 arrays with the size of $(C + K - 1)$, denoted by $routes$ and $trips$. The $routes$ represents the paths of the shippers by storing the ids of C customers and $(K - 1)$ shippers and arrange in random order. Positive integers used to represent the customer ids and negative integers used to present the shipper ids. $trips$ used to identify the trips of the shippers. **Figure 4** shows the chronosome representation for an example of 12 customers with ids from 1 to 12, 3 shippers with ids 1 to 3, 2 warehouses A and B. In the figure, first three elements of $routes$ are 12,2 and 5. It means the shipper with id 1 is assigned to deliver to these customers. The corresponding elements in $trips$ are binary only. 0 means shipper can directly continue to travel, meanwhile 1 means shipper has to comeback related warehouse to load the package before devering to the next customer. $trips$ stores only $K - 1$, in this case shipper id 1 is not nessessary stored in the $routes$ for more convinent in ramdom process.

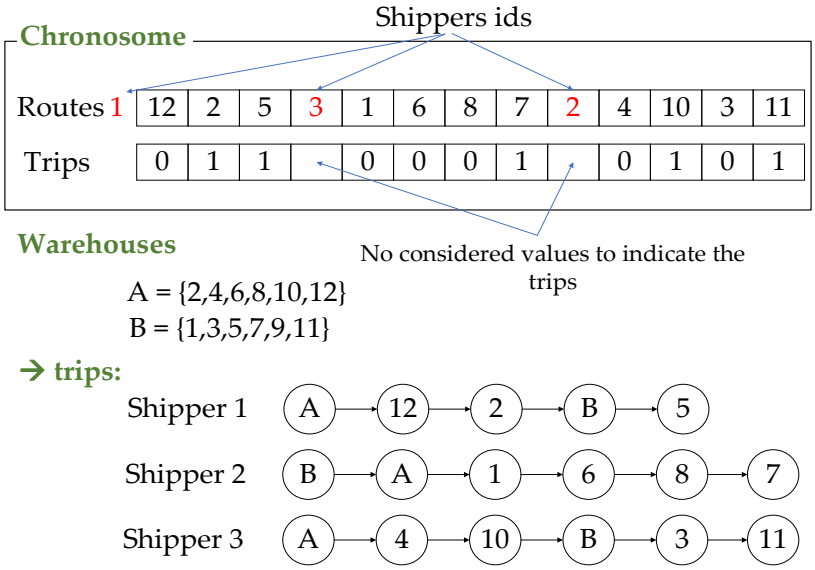

Figure 4: Chromosome representation.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

2. Fitness function: we used the compromise euclidean distance based fucntion for the individual as:

$$p.fitness = \left(\sum_{i=1}^{4} w_i \left|\frac{F_i - z_i^*}{z_i^{worst} - z_i^*}\right|^2\right)^{1/2} \qquad \forall\, p \in P$$

All proposed algorithms give optimal results for single-objective optimization problems. z_i^* and z_i^{worst} can be considered as pre-computed in this procedure.

3. Selection: we chose φ elite individuals and bypassed them from the crossover and mutation phase to keep them in the next generation.
4. Crossover: creates a new solution that retains the good properties of its parent. We select a crossover rate μ. There are 5 steps to implement the crossover for the whole remaining individuals of the next generation (see **Figure 5**Error! Reference source not found.) as following:

   Step 1: select randomly 2 individuals as the parents denoted by p_1, p_2.

   Step 2: Randomly select a substring from a parent for both $routes$ and $trips$.

   Step 3: Create a proto-child by phase the substring into its corresponding position.

   Step 4: Delete the all the elements that already in the proto-child of the remand parents. This creates an array that have the elements that proto-child need.

   Step 5:

   - For $routes$: Place the elements of the result array into unfix position of proto-child from left to right.
   - For $trips$: Place the elements of the result array into unfix position of proto-child in the corresponding position.
5. Mutation: Modify a solution to create a new solution to expand the search space of the algorithm. We select a mutation rate ω. There are two steps to implement the mutation for the whole remaining individuals of the next generation as follows:

   Step 1: Randomly select a substring from the individual.

   Step 2:

   - For $routes$: shuffle the element in the substring to create a new route.
   - For $trips$: flip each element in the substring to create a new trip.

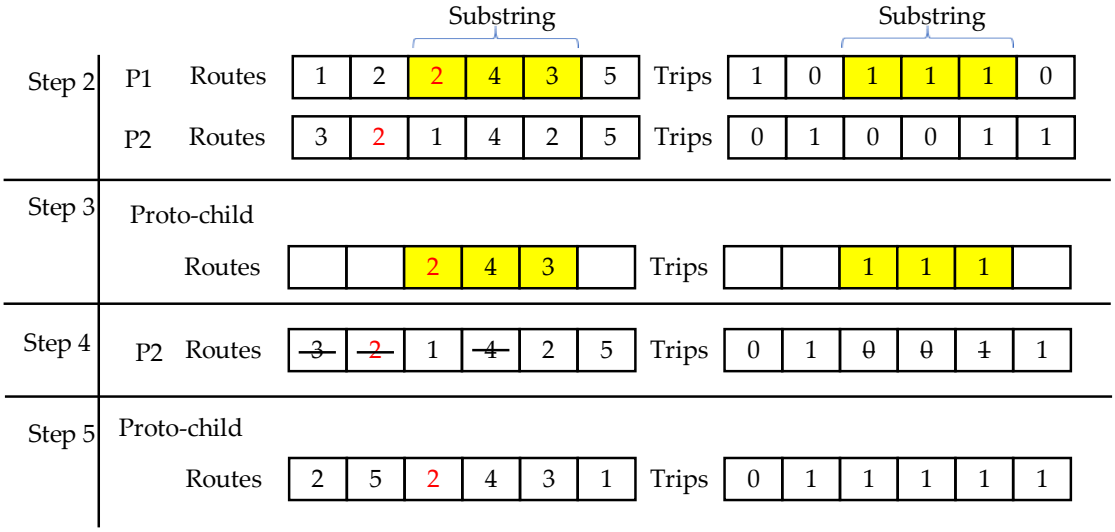

Figure 5: step 2 to step 5 of the Crossover phase.

6. Repair: In this phase we fix the solutions that violate the defined constraints. There are some principal rules to guide the repairing process:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

- The $trips$ array controls the trips of the shippers that are related to the capacity. If the customers' weight is already surpassed the corresponding shipper in a journey, the $trips$ array must be fixed for that shipper to go back for supply after the current customer.
- We keep the principal to minimize number of trips, therefore we check the *trips* if there is any $trips[i] = 1$ that can be removed without violate the capacity constraint and remove it if possible.
- At the end of each trip, the corresponding element value in the $trips$ array must be 1.
- $trips[i] = 1 \ \forall \ i = 1 \dots (C + K - 1) \ \wedge \ routes[i] < 0$.

3.2. Hybrid Genetic Algorithm

3.2.1. Local Search

Local search [**48**] is an algorithm using a single search path (searching in the neighborhood) to improve the initial solution for a better solution. The solution point is structured the same as the chromosome representation of GA as described in the section 3.1. The process of the Local search can be described in 2 steps as follows:

1. Denote s is starting solution.
2. Find $S = searchNeighborhood(s, k)$ is the set of neighbor's solutions of s.
   Where: k is the size of S.
   $searchNeighborhood(s, k)$ function to return k neighbor's solutions.
3. Repair every solution s' in S that violates the defined constraints.
4. Return s* = $\underset{s' \in S}{\text{argmax}}(s'.fitness)$.

3.2.1. Combination of GA and Local Search

One risk for GA is that individuals can become trapped in local optima, often caused by designs that fail to maintain population diversity or an insufficient number of search agents. In this study, we take advantage of the better neighborhood search feature of Local Search to give individuals a better chance of overcoming the stuck in Local Optima. We run the Local Search several times corresponding to the elite individuals obtained by GA as starting points to retrieve better solutions. These solutions in the next generation then replace the inputs. The process is illustrated in **Figure 6**.

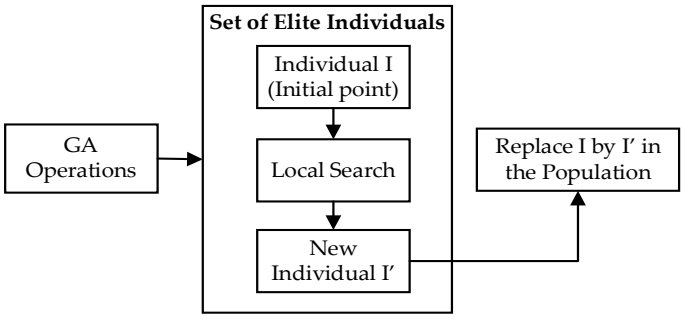

Figure 6: Combination of GA at g^{th} generation and the Local Search in HGA.

3.3. Tabu Search

Tabu search is an improved version of the Local Search used for mathematical optimization [**49**]. Local Search methods tend to get stuck in suboptimal regions. TA enhances the performance of these techniques by banning accessed solutions or other solutions through user-supplied rules. We implement the principal mechanism of TA and reuse the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

data structure and algorithms described in the previous parts. The flow of TA is illustrated in **Figure 7**.

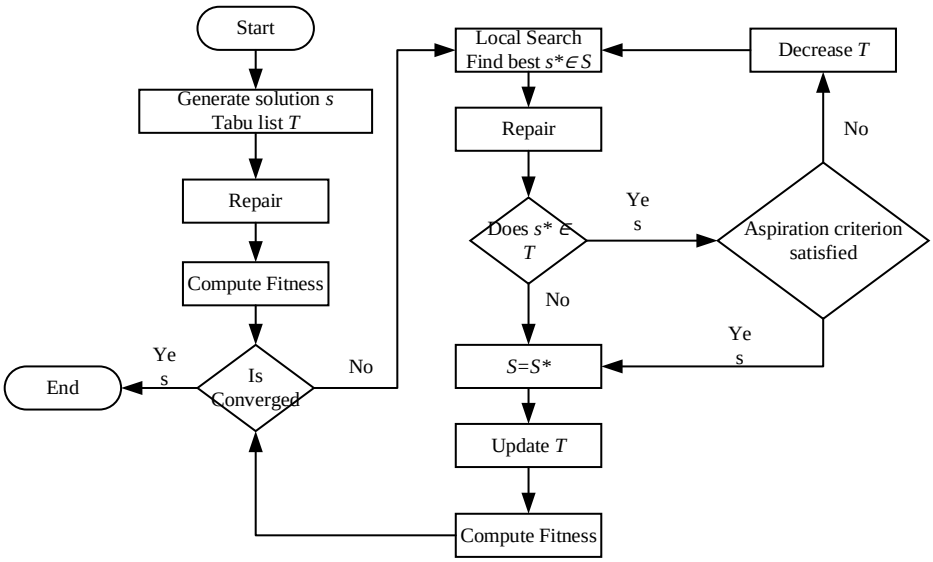

Figure 7: The flow of TA

4. Experimental Results

4.1. Experimental Design

To evaluate the effectiveness of the proposed method, we use a collected dataset in one business day received from the shipment company in Hanoi, Vietnam. It consists of 200 orders, distributed from 5 warehouses, orders delivered by ten shippers. Customer locations are collected via GPS. To avoid detailed measurements in the scheduling process, the company transformed the customer's precise coordinates to the center of the street. Travel time, the average speed of shippers on given time is measured based on collected data from google map and check-in data of the shippers. **Figure 8** illustrates the overview of the experimental design.

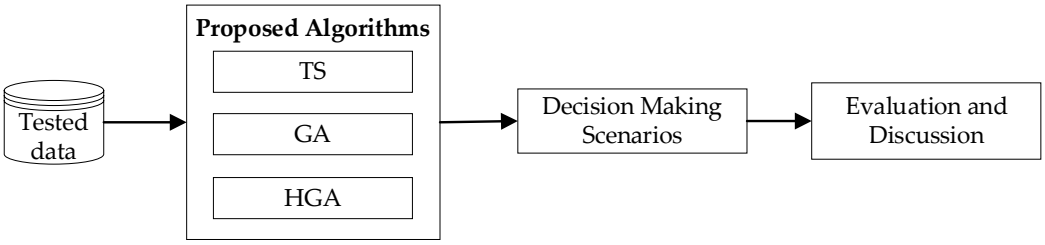

Figure 8: Overview of Experiment Design

We conducted experiments and analyzed the results of three proposed algorithms, including TA, GA, and HGA, in terms of convergence, processing speed, and solution quality. Then, the best-performing algorithm is selected for testing with different decision scenarios. The experiments are implemented in the computer with detailed configuration, as shown in **Table 2**.

Table 2: System configurations for experiments

Item	Info	
CPU	Intel(R) Core (TM) i5-8350U CPU @ 1.70GHz	1.90 GHz
RAM	Corsair Vengeance LPX 8GB	
Programming Platform	java 8	

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

Operating System	Window 10

Parameters govern metaheuristic algorithms. We have tested several different values. Each of them can affect both the computational cost as well as the quality of the solution. For example, the more search agents have been used, the higher the chance of finding the global optima, but the search time of each agent is also generated. It significantly increases the computation time. The following experiments are performed with the appropriate settings to highlight the performance of each designed algorithm, as shown in **Table 3**.

Table 3: Parameters to conduct the experiments.

Parameter	GA	HGA	TA
Population size	1000	100	1
Crossover rate	0.9	0.9	None
Mutation rate	0.3	0.3	None
Selection rate	0.1	0.1	None
Stop condition	100	100	100
Neighborhood structure	None	Replace	Replace
Scanned Neighbors	None	1000	-
Tabu tenure	None	None	3

4.2. Results

As mentioned in section 2.2, the original objective functions are transformed to the distance function using compromise programming. We solve the problem as separated single-objective problems. The worst point is identified in the same way as the ideal point but using the max function for the objective function. The ideal point and worst point are shown in **Table 4**. Three designed algorithms give the same result when solving these single-objective problems.

Table 4: Obtained results by solving the problem as separated single objective problems

i	z_i^*	z_i^{worst}
1	699.32	5045.18
2	0	701979.5
3	0	12137.6
4	1.35	9582.21

The detailed solution of each Single Objective problem is described in **Figure 9.** In the first case, we try to maintain the lowest transport costs. The system only needs to use 7 out of 10 shippers, as shown in **Figure 9A**. However, the late delivery time is considerable (181801.5), the time the shippers wait and the difference in workload is 2705.85 and 2274.8 time-units, respectively. All shippers are mobilized to deliver on time (**Figure 9B**). However, the transport cost also increased to 2438.09. To avoid late delivery, shippers must arrive earlier than the scheduled time for several orders then wait until the right time to deliver. The total waiting time is 6894.85. The difference in workload is also relatively significant when the shipper with the highest workload has to work more than 257,875 time-units to the average. **Figure 9C** and **Figure 9D** show the results when the scheduler optimizes the objectives f3 and f4 as respectively $(f_1 = 2676.491; f_2 = 152876.1; f_3 = 0; f_4 = 2649.96)$,$(f_1 = 2734.456018; f_2 = 99640.8; f_3 = 6635.5; f_4 = 1.35)$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

k	O_k	f_1
1		0
2	2-1-3-4-117-63-93-80-85-107-60-7-56-162-71-32-145-40-79-189-42-41	77.39062
3	5-4-1-2-36-179-125-161-101-115-193-127-100-191-148-109-62-129-167-49-102-44-18-29-144-174-57-176-6-47-73-99-5-4-3-2-1-197-154-124-120-30-34-131-69-76-97-163-26-111-22-87-195-205-24-182-21-184-16-177-27-103-11-14-178-175-121-64-188-126-151-190-2-3-4-5-1-204-98-45-54-166-186-55-130-48-138-61-140-139-137-108-196-201-20-156-142-43-147-116-187-106-168-172-86-159-183-74-28-78-2-1-4-3-83-119-165-12-112-143-113-66-104-77-171-25-75-8-15-157-19-122-17-192-185-23-72-70-58-68-173-2-1-3-4-5-155-37-50-39-89-38-59-164-136-150-81-53-82-84-13-170-33-65-46-194-31-206-203-114-158-2-3-4-1-160-92-153-132-91-35-123-199-67-200	542.294
4	5-118-202	10.79831
5		0
6	2-198	1.365256
7	3-4-2-10-134-141-180-169-133-152-128-146-110-94	41.42976
8	3-9	3.066012
9	2-4-3-52-149-181-96-90-88-105-51	22.97737
10		0
	Total:	699.3214

A

k	O_k	f_2
1	5-1-2-31-30-130-121-78-21-55-71-64-3-4-122-133	0
2	1-3-4-72-158-192-88-54-28	0
3	1-2-7-77-92-1-2-5-22-62-29	0
4	2-34-4-3-2-1-5-23-200-52-9-81-41-168-193-96-11-35-20-155-97-132-104-150	0
5	4-2-102-37-3-4-2-1-186-141-59-179-198-173-103-188-115-24-172	0
6	1-3-4-2-5-203-139-124-85-145-46-16-38-111-8-176-26-67-163-116-89-3-5-2-140-36-182-185-100-61	0
7	5-1-3-4-2-44-174-157-10-181-175-49-53-57-75-12-51-114-43-162-129-14-93-98-126-94-66-47-167-33-113-152-3-4-32-154	0
8	1-2-5-3-4-161-110-69-143-76-171-149-63-164-142-165-84-120-196-118-151-177-83-82-127-169-125-205-148-160-128-119-65-4-5-3-195-27-87-79-45-5-170	0
9	3-4-5-2-1-15-91-191-25-153-166-204-159-17-58-183-147-90-18-106-190-136-73-117-105-138-144-109-202-146-13-180-5-99	0
10	2-5-4-1-3-156-178-131-187-189-48-56-101-6-199-108-40-123-60-197-194-184-74-39-134-137-42-70-50-68-107-19-206-112-3-5-80-86-4-201	0
	Total:	0

B

k	O_k	f_3
1	4-5-146-163-31-54-167-4-5-2-3-1-180-138-100-205-92-183-186-62-121-99	0
2	3-12	0
3	2-3-4-1-5-185-15-173-147-58-88-9-145-18-111-161-60-46-118-70-83-201-8-56-116-20-63-96-109-93-29-169-2-4-5-1-3-37-192-149-168-24-154-123-94-26-23-112-164-152-206-77-97-21-50-148-59-181-45-79-91-57-141-69-4-1-5-3-2-165-119-178-98-25-157-89-134-40-108-19-75-74-14-128-199-175-137-27-184-85-142-86-196-38-33-171-162-202-3-39-158	0
4	5-193-5-2-166-151	0
5	4-2-133-53-4-2-1-5-3-104-52-187-13-179-41-6-78-115-67-153-51-82-16-177-204-73-17-61-150-84	0
6	2-129-3-76	0
7	3-80-2-4-3-5-1-155-131-124-170-127-87-143-32-200-101-113-189-68-125-64-43-188-120-66-107	0
8	3-2-4-1-5-122-198-194-114-139-90-136-44-126-195-30-65-49-172-34-159-7-160-102-110-71-35-174-132-106-47-81-1-2-22-117	0
9		0
10	2-1-4-3-5-182-72-11-156-42-203-48-130-55-10-176-36-105-191-197-103-28-190-144-3-140	0
	Total:	0

C

k	O_k	f_4
1	1-4-2-3-5-172-8-52-153-147-25-24-44-2-5-3-57-100-202-196	0.045
2	4-5-2-1-3-23-86-182-123-137-94-9-50-39-114-67-13-155-166-128-126-204-5-3-170-80	-0.005
3	5-1-4-3-178-99-75-101-174-122-4-3-5-1-2-201-10-30-157-115-161-15-103-32-76-37-141-158-74	-0.155
4	3-2-1-5-56-185-35-121-163-17-109-118-51-45-108-159-5-2-130-31-127-62	-0.405
5	2-4-5-1-198-142-27-26-160-20-53-191-48-92-165-193-34-168-180-149-72-116-4-3-179-200	0.145
6	4-1-3-2-5-132-119-77-183-113-192-7-68-12-156-139-176-61-173-29-14-43-205-138	0.295
7	3-2-1-5-4-42-19-6-120-40-194-188-104-144-63-129-82-151-83-107-2-5-4-33-69-105	0.145
8	5-1-4-2-3-78-21-177-88-134-117-125-186-140-91-97-55-84-66-150-71-58-162-79-124-59-171-81-146-110-106-54-85-148	0.045
9	4-2-5-1-3-102-184-46-195-70-154-90-203-36-73-167-38-145-143-11-189-164-112-4-5-175-64	-0.105
10	4-2-5-3-1-49-199-152-111-87-41-197-169-93-60-47-18-190-96-131-22-65-206-89-133-136-28-181-1-5-3-187-16-98	-0.005
	Total:	1.35

D

Figure 9: Generated travelling paths for shippers by solving single objective problems. A) f_1; B) f_2; C) f_3; D) f_4;

To compare algorithms, we set the weight parameters to be the same. Although theoretically, multi-objective optimization may not exist as the best solution. When other solutions do not entirely dominate other solutions, we base on the obtained values of the distance function (objective values) to rank the solutions. The metaheuristics operations are stochastic. Therefore, to evaluate the stability of the proposed algorithms, we execute the 15 times and the obtained results are shown in **Table 5.** The numbers show that GA-based algorithms show that they can receive quality results at a much smaller cost than TA. Using multiple search agents in TA, each of which continues to search for quality neighbors, is a computationally expensive process. To achieve a similar solution quality as HGA, the average processing time of TA is 2.06 times more enormous on the tested dataset. We normalize the objective values in the range [0,1] based on obtained values in **Table 4**. We only use a single core to executes the algorithms. The executions can be speeded up using parallel mechanism for search agents proposed by Ngo et al [**44**].

Table 5: Best obtained results by proposed algorithms

Algorithm	Solution Quality							Average Time(min)
	Average Fitness	Best solution					Worst Fitness	
		Fitness	f_1	f_2	f_3	f_4		
TA	0.052	0.048	0.076	0.079	0.021	0.027	0.055	25.4
GA	0.073	0.068	0.041	0.096	0.013	0.057	0.076	5.35
HGA	0.050	0.045	0.047	0.068	0.016	0.034	0.053	12.6

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

Both GA and HGA algorithms use a similar search mechanism. The only difference is that HGA continued to use local search to find neighbors with better fitness values before creating a new generation. Theoretically, this ensures that the HGA has a better chance to get over the local optima than the original version. It has also been confirmed through our experiments. However, because many individuals must perform local search operations after genetic operations, the total time to search for solutions for each increased significantly. However, HGA can provide high-quality solutions when the obtained solution completely dominates the solutions of original GA and is slightly better than TA on the tested dataset of 200 customers.

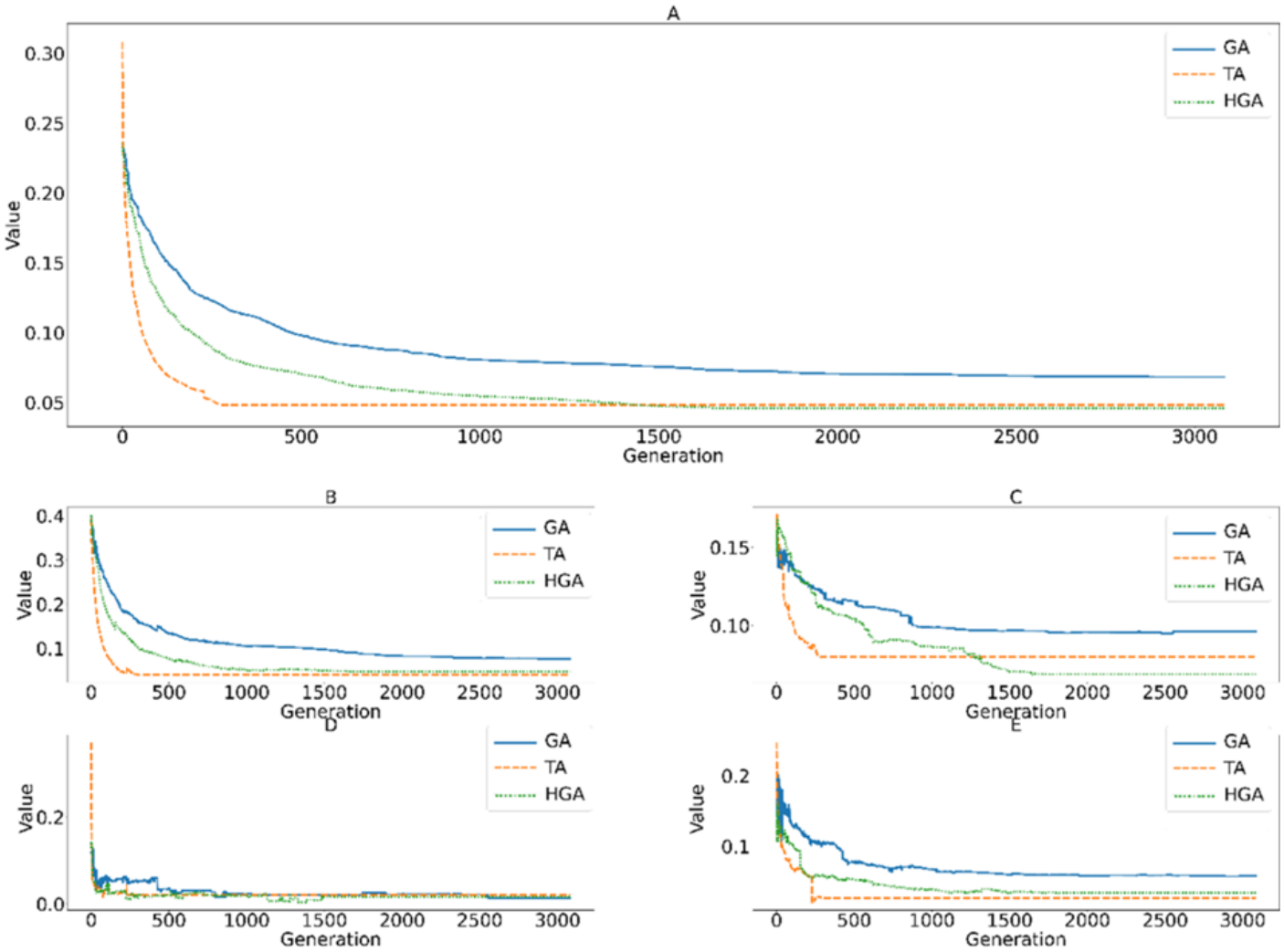

Figure 10 A) Fitness Values; B) $f_1(0)$; C) $f_2(0)$; D) $f_3(0)$;E) $f_4(0)$; of designed algorithms change over generations/iterations.

The change of fitness values can visualize the convergence of the algorithms through each generation/iteration in **Figure 10**. In the figure, we show all algorithms running up to 3000 iterations for convenience in comparison. However, these algorithms still respected their stop conditions. The result mentioned in the previous section is the time to reach the final solution. The change of fitness values shows that TA has obtained better results in the first few iterations than GA-based algorithms. However, up to the 297th iteration with a fitness value of 0.048 is a local solution that TA cannot pass. Meanwhile, the GA and HGA algorithms show that they have maintained the population diversity as the next generations continue to improve the quality of the solution. HGA has provided solutions that have been continuously improved over the generations. Until it found the final solution (0.046) at 1849-th generation. This continuous improvement is significant in practice. The algorithmic stopping condition that can be determined by the number of generations with the same result is minor to avoid the computational cost. The different

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

objective functions may increase at some generations, but they are decreased in general because the algorithm consistently reduces fitness values.

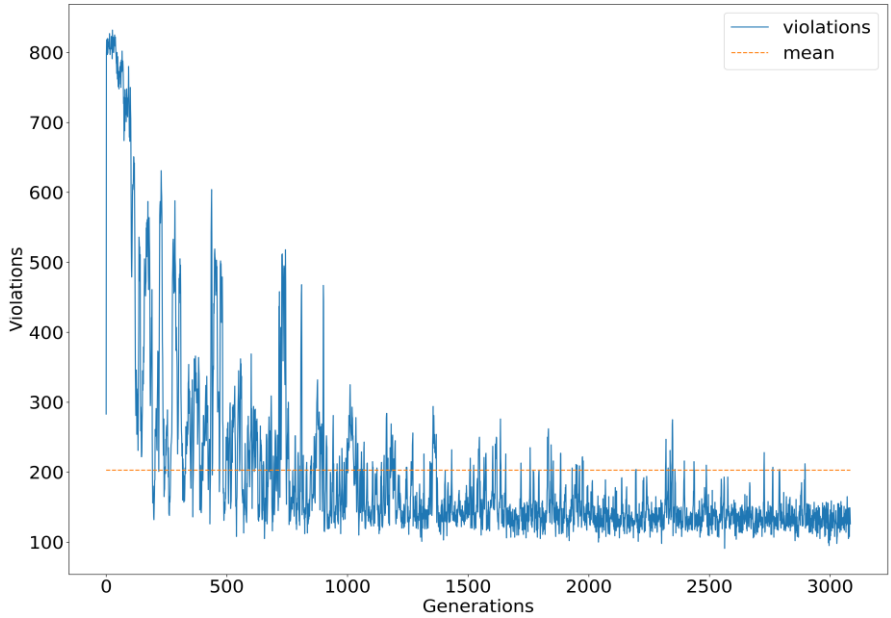

Figure 11: Number of violated constraints with corresponding iterations of the search process of HGA.

The stochastic mechanism for generating solutions generates a series of solutions that violate constraints. In some cases [**45**], hose solutions can be eliminated by the searching process. We must have a mechanism to correct the error solutions in this problem. This process speeds up the algorithm through the acquisition of valid answers. The number of invalid solutions decreases after newly generated solutions. For example, for the GA-based algorithms, unviolated constraints parent solutions genes are selected and crossed. However, the mutation process produces a certain number of invalid solutions. **Figure 11** displays the number of violated constraints with corresponding iterations of the search process of HGA. The data distribution affects the reduction of values in the distance-based fitness function. For example, the value of objective function f_4, the Workload of the shipper, seems to have played a more significant role than the dense distribution of the values in the objective function f_1, as observed from the generated solution by HGA in **Table 6**. However, the search operations can be directed by calibrating the weight parameters.

Table 6: generated traveling paths 10 shippers to deliver 200 packages from 5 warehouses by HGA.

k	O_k	f_1	f_2	f_3	f_4
1	3-28-4-3-2-1-102-124-148-154-110-146-123-132-94-137-115-204-205-203-114-158	107.9532	8350.7	4.4	0.255
2	1-2-4-3-5-60-184-188-198-51-52-162-141-75-40-93-143-92-128-160-67-199-200-29-108-18-66-96-113-171-25-138-44-35-71-179-2-3-152-32	125.272	5687.2	57.3	141.555
3	5-4-2-202-126-151-147-107-142-156-190-193-127-100-30-145-112-165	96.75424	6272.4	1.9	-6.895
4	5-4-1-3-43-173-125-78-22-6-56-176-81-169-76	78.75142	973.15	0	-37.845
5	5-2-3-4-1-170-13-79-55-20-185-19-129-15-195-197-24-182-109-136-164-150-133-65-27-201-33-180-99-80-50	110.138	4980.6	0	-0.395
6	3-9-4-1-175-7-38-59	24.40988	149	0	-118.295
7	2-4-3-1-5-117-57-134-103-11-14-192-42-41-161-186-166-62-72-23-177-16-194-31-183-10-46-74-58-68	101.6263	7904.1	0	0.055
8	5-1-4-3-2-36-187-105-89-39-90-88-116-206-106-159-86-172-168-155-163-12-97	89.68427	3929.95	28.3	-0.845
9	5-4-3-2-1-178-101-149-181-130-84-82-48-8-191-49-157-189-21-34-77-120-153-91-174-53-131-63-69-37-85-118	87.14513	4240.15	92.3	22.205
10	5-1-2-4-3-87-47-83-73-111-26-119-70-121-64-17-122-167-61-140-104-98-45-54-139-144-196	84.42806	5860.5	14.4	0.205

To evaluate the adaptability of the algorithms to different scales of the system, we divided the tested dataset into smaller datasets with 50,100,150,200 customers respectively

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

to conduct experiments, as shown in **Figure 12**. The processing speed of TA slows down proportionally to the system scales. The quality of HGA is slightly better than TA and significantly better compared to the original GA. HGA's processing time is growing faster than GA but better compared to TA. HGA and TA both use neighbor searching, but genetic operations seem to be more effective to identify initial points before search for neighbor points than TA's hill-climbing mechanism.

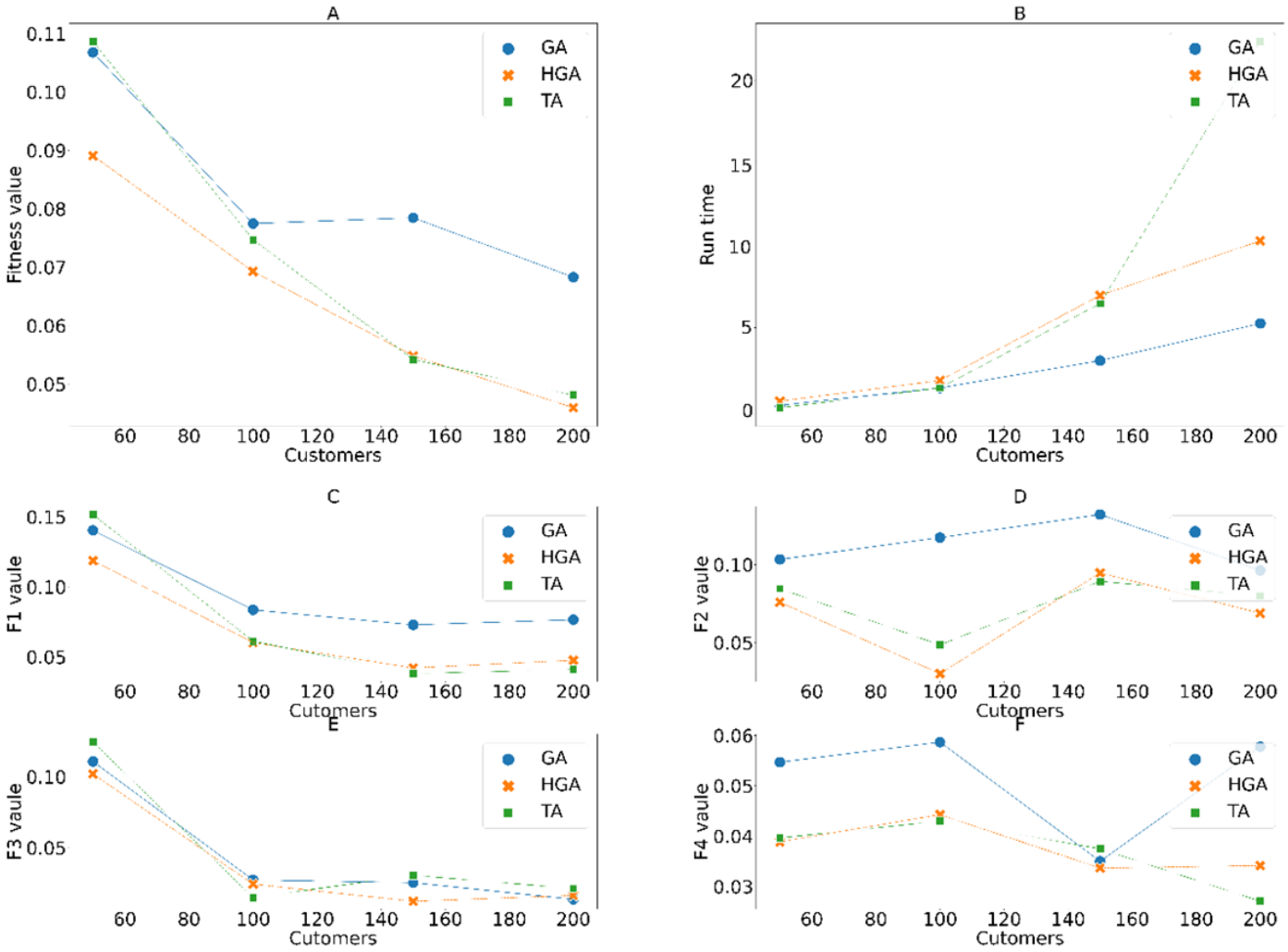

Figure 12: A) Fitness Values; B) Execution Time; C) f_1; D) f_2; E) f_3; F) f_4; obtained with different number of customers to serve.

Approaches to the MOP problem based on the decomposition of multi-objective functions to single-objective functions have many advantages. Compromise Programming is a suitable solution when the decision-maker cannot assign preferences for each specific goal. They have a weakness that is very difficult to illustrate Pareto Frontier. However, through weight parameters, decision-makers can experiment with different decision criteria. We compared solutions generated by the proposed algorithms. These solutions do not fully dominate (all objective values are better) each other. Therefore, to evaluate which algorithm performs better in different decision-making situations. In this experiment, we selected the sub-dataset of 100 customers then obtained ten solution points corresponding to different values of weight parameters for each algorithm, as shown in **Figure 13**. We then calculate the Hypervolume HVC [**50**] for the solutions obtained by the algorithm as follows:

$$HVC = \frac{volume(\bigcup_{s \in S}(s, z^{worst}))}{volume(cube(z^*, z^{worst}))}$$

Where:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

- s is the solution in the Pareto solutions set S that generated by the algorithm.
- $cube(a,b)$ denotes the oriented axes hypercube that formulated by points a and b in the objective space.
- $volume(c)$ denotes the volume of the hypercube c in the objective space.

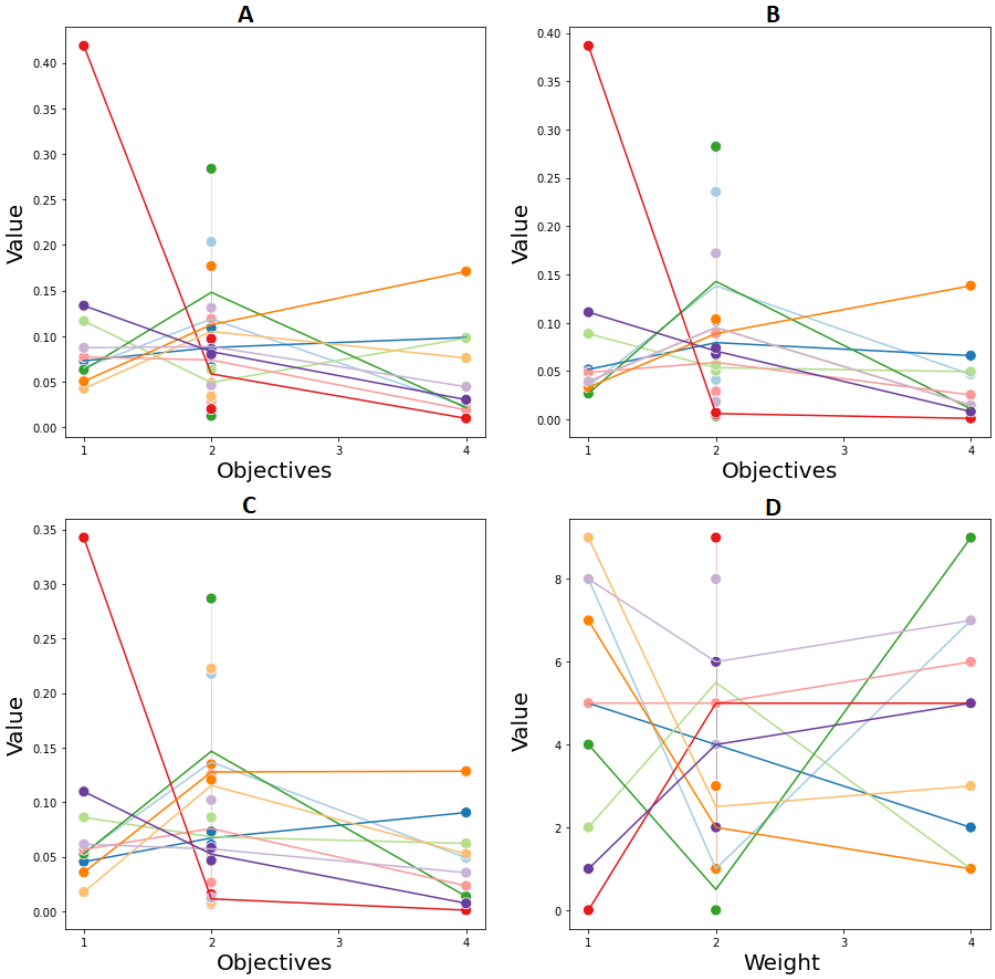

Figure 13: 10 obtained solutions in 4D objective space by A)GA, B)TA, C)HGA with different of weight parameters (D).

The results listed in **Table 7** show that the HGA's hypervolume is similar to TA and better than the GA. The larger the HVC value, the closer the algorithm can discover solutions close to the actual Pareto frontier. Through TA's nearest neighbor search, the hill-climbing mechanism allows it to overcome the local optimal better than the original GA. However, GA can effectively integrate with other methods to improve quality without trading enormous computational costs. The hybrid version of EA shows its effectiveness in different decision-making scenarios.

Table 7: Best obtained results by proposed algorithms

Algorithm	HVC
TA	0.938
GA	0.885
HGA	0.941

To evaluate the capabilities of the proposed CP-based method. We use genetic operations designed to implement a version of the NSGA-2 algorithm [**51**]. The parameters to execute algorithm and the obtained results on dataset of 200 customers are shown in **Error! Reference source not found.**. This setup is to make NGSA-2 can find the best (possibly)

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 January 2022 doi:10.20944/preprints202201.0130.v1

values of each objective function f_1, f_2, f_3, and f_4. NGSA-2 shows its power to search for a Pareto front with more than 8000 solutions after more than 6 hours of execution. The NGSA-2 can archive the solutions with best values of $f_2 = 0$ and $f_3 = 0$, which is similar to proposed algorithms; however, the proposed method shows it to be completely superior when looking for solutions with f_1 and f_4 being better. The normalized distance of closest solution to z^* (fitness value) obtained by NGSA-2 is 0.122 that is inferior to compare with the generated one by CP-based GA when using the similar searching mechanism. Although our results are not enough to conclude that the CP-based method is better than MOEA-2, the obtained Pareto front may contain lower quality solutions to the proposed method. The effort to search for the Pareto frontier leads the search agents not to focus on achieving their goal as SOP. It requires a significant computational overhead, which is difficult to adapt in a real-world environment. The user has no other choice, even if they only need to use one solution in reality. Other factors in the decision problem, such as user experience, contribute nothing to this centralized search effort.

Table 8: Obtained results on the tested dataset by NGSA-2.

Parameter / Criteria	Applied/Obtained by NGSA-2	Applied/Obtained by CP-based GA
Population	10000	1000
Stop Condition	1000	100
Crossover rate	0.8	0.9
Mutation rate	0.3	0.3
Average Execution Time (min)	~372	~5
Number of Solutions	~8837	1
Best found f_1	1422	699.32
Best found f_2	0	0
Best found f_3	0	0
Best found f_4	42	1.35
Best fitness value	0.122	0.0689

5. Conclusions

This study presents an adaptive method to solve the urban shipment problem as MOP-VRP based on CP and Metaheuristics. The proposed model is a new variant of the VRP problem that combines different types of VRP and MOP where terrain and traffic conditions over time are integrated. We also designed three algorithms, GA, HGA, and TA, to solve the proposed model and compare their performance on the tested dataset. Combining compromise programming and metaheuristics is suitable for approaching the MOP problem. However, once this approach is chosen, the decision-making process needs to respect compromise solutions instead of finding the Pareto frontier and assigning a solution based on higher-level information like other approaches such as Pareto-dominance-based MOEA. In return, this approach allows flexible design for many business scenarios. Traditional metaheuristics methods or hybrid versions are smoothly applied with the CP-based. Although, the CP-based system introduced weight parameters to the objective functions. The selection of these values in practice depends heavily on the decision-makers experience and the business sense using the model because re-executing the algorithm with large datasets multiple times leads to a prohibitive computation cost. Therefore, it is recommended as an option when the decision-maker does not have sources to indicate the preferences that happen more often in practice. The test results show that the combination of GA and Local Search in HGA creates a superior advantage in improving the quality of the solution. The original version of GA may use trivially sampling points, but the nearest neighbor search can provide better genes to the next generation. This combination produces a high-quality solution without trading off too much compu-

tational cost like the nearest neighbor search with a memory mechanism in TA. Our upcoming work is to integrate the VRP model with integral logistics problems. The improvement of the algorithm using recent advances in metaheuristics is the priority.

Author Contributions: Conceptualization, S.T.N. J.J., and I.A.A.; methodology, S.T.N., J.J. and I.A.A.; software, S.T.N. and G.H.N.; validation, S.T.N., and G.H.N.; formal analysis, S.T.N., J.J. and I.A.A.; investigation, S.T.N., G.H.N., A.N.B. and M.U.A.; resources, S.T.N., G.H.N., A.N.B. and M.U.A.; data curation, S.T.N. and G.H.N.; writing—original draft preparation, S.T.N.; writing—review and editing, S.T.N., J.J. and I.A.A.; visualization, S.T.N. and G.H.N.; supervision, J.J., I.A.A. and A.N.B.; project administration, S.T.N. and M.U.A.; funding acquisition, S.T.N. and A.N.B.

Funding: This research was funded by FPT University, under decision number QĐ1097/QĐ-ĐHFPT and QĐ 1393/QĐ-ĐHFPT for project number HO-CPDT2021. Article Processing Charge was funded by Universiti Teknologi PETRONAS.

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

References

1. P. Toth and D. Vigo, Eds., The Vehicle Routing Problem. Society for Industrial and Applied Mathematics, 2002.
2. A. Mor and M. G. Speranza, "Vehicle routing problems over time: a survey," 4OR, vol. 18, no. 2, pp. 129–149, Jun. 2020, doi: 10.1007/s10288-020-00433-2.
3. S. Kır, H. R. Yazgan, and E. Tüncel, "A novel heuristic algorithm for capacitated vehicle routing problem," J. Ind. Eng. Int., vol. 13, no. 3, pp. 323–330, Sep. 2017, doi: 10.1007/s40092-017-0187-9.
4. M. C. Bouzid, H. Aït Haddadene, and S. Salhi, "An integration of Lagrangian split and VNS: The case of the capacitated vehicle routing problem," Comput. Oper. Res., vol. 78, pp. 513–525, Feb. 2017, doi: 10.1016/j.cor.2016.02.009.
5. D. S. W. Lai, O. Caliskan Demirag, and J. M. Y. Leung, "A tabu search heuristic for the heterogeneous vehicle routing problem on a multigraph," Transp. Res. Part E Logist. Transp. Rev., vol. 86, pp. 32–52, Feb. 2016, doi: 10.1016/j.tre.2015.12.001.
6. V. F. Yu, A. A. N. P. Redi, Y. A. Hidayat, and O. J. Wibowo, "A simulated annealing heuristic for the hybrid vehicle routing problem," Appl. Soft Comput., vol. 53, pp. 119–132, Apr. 2017, doi: 10.1016/j.asoc.2016.12.027.
7. Alemany G., Juan A.A., Garcia R., Garcia A., Ortega M. (2018) Multi-capacity, Multi-depot, Multi-product VRP with Heterogeneous Fleets and Demand Exceeding Depot Capacity. In: Gil-Lafuente A., Merigó J., Dass B., Verma R. (eds) Applied Mathematics and Computational Intelligence. FIM 2015. Advances in Intelligent Systems and Computing, vol 730. Springer, Cham. https://doi.org/10.1007/978-3-319-75792-6_10.
8. J. R. Montoya-Torres, J. López Franco, S. Nieto Isaza, H. Felizzola Jiménez, and N. Herazo-Padilla, "A literature review on the vehicle routing problem with multiple depots," Comput. Ind. Eng., vol. 79, pp. 115–129, Jan. 2015, doi: 10.1016/j.cie.2014.10.029.
9. L. Zhen, C. Ma, K. Wang, L. Xiao, and W. Zhang, "Multi-depot multi-trip vehicle routing problem with time windows and release dates," Transp. Res. Part E Logist. Transp. Rev., vol. 135, Mar. 2020, doi: 10.1016/j.tre.2020.101866.
10. E. Babaee Tirkolaee, P. Abbasian, M. Soltani, and S. A. Ghaffarian, "Developing an applied algorithm for multi-trip vehicle routing problem with time windows in urban waste collection: A case study," Waste Manag. Res. J. a Sustain. Circ. Econ., vol. 37, no. 1_suppl, pp. 4–13, Jan. 2019, doi: 10.1177/0734242X18807001.
11. D. Cattaruzza, N. Absi, and D. Feillet, "Vehicle routing problems with multiple trips," 4OR, vol. 14, no. 3, Sep. 2016, doi: 10.1007/s10288-016-0306-2.
12. W. Li, Y. Wu, P. N. R. Kumar, and K. Li, "Multi-trip vehicle routing problem with order release time," Eng. Optim., vol. 52, no. 8, Aug. 2020, doi: 10.1080/0305215X.2019.1642880.
13. B. Pan, Z. Zhang, and A. Lim, "Multi-trip time-dependent vehicle routing problem with time windows," Eur. J. Oper. Res., vol. 291, no. 1, pp. 218–231, May 2021, doi: 10.1016/j.ejor.2020.09.022.
14. Ma Y., Han J., Kang K., Yan F. (2017) An Improved ACO for the Multi-depot Vehicle Routing Problem with Time Windows. In: Xu J., Hajiyev A., Nickel S., Gen M. (eds) Proceedings of the Tenth International Conference on Management Science and Engineering Management. Advances in Intelligent Systems and Computing, vol 502. Springer, Singapore. https://doi.org/10.1007/978-981-10-1837-4_96.
15. W. Zhang, Y. Gajpal, S. S. Appadoo, and Q. Wei, "Multi-Depot Green Vehicle Routing Problem to Minimize Carbon Emissions," Sustainability, vol. 12, no. 8, p. 3500, Apr. 2020, doi: 10.3390/su12083500.
16. S. Nucamendi-Guillén, A. Gómez Padilla, E. Olivares-Benitez, and J. M. Moreno-Vega, "The multi-depot open location routing problem with a heterogeneous fixed fleet," Expert Syst. Appl., vol. 165, p. 113846, Mar. 2021, doi: 10.1016/j.eswa.2020.113846.
17. B. C. Shelbourne, M. Battarra, and C. N. Potts, "The Vehicle Routing Problem with Release and Due Dates," INFORMS J. Comput., vol. 29, no. 4, Nov. 2017, doi: 10.1287/ijoc.2017.0756.
18. M. Han and Y. Wang, "A Survey for Vehicle Routing Problems and Its Derivatives," IOP Conf. Ser. Mater. Sci. Eng., vol. 452, p. 042024, Dec. 2018, doi: 10.1088/1757-899X/452/4/042024.
19. R. Ojstersek, M. Brezocnik, and B. Buchmeister, "Multi-objective optimization of production scheduling with evolutionary computation: A review," Int. J. Ind. Eng. Comput., pp. 359–376, 2020, doi: 10.5267/j.ijiec.2020.1.003.

20. J. Dutta et al., "MULTI-OBJECTIVE GREEN MIXED VEHICLE ROUTING PROBLEM UNDER ROUGH ENVIRONMENT," Transport, pp. 1–13, Feb. 2021, doi: 10.3846/transport.2021.14464.
21. O. Bahri, N. Ben Amor, and E.-G. Talbi, "Robust Routes for the Fuzzy Multi-objective Vehicle Routing Problem," IFAC-PapersOnLine, vol. 49, no. 12, pp. 769–774, 2016, doi: 10.1016/j.ifacol.2016.07.867.
22. Gunantara, N. (2018). A review of multi-objective optimization: Methods and its applications. Cogent Engineering, 5(1). https://doi.org/10.1080/23311916.2018.1502242.
23. Okabe, T., Yaochu Jin, & Sendhoff, B. (n.d.). A critical survey of performance indices for multi-objective optimisation. The 2003 Congress on Evolutionary Computation, 2003. CEC '03. https://doi.org/10.1109/CEC.2003.1299759.
24. J. Lysgaard and S. Wøhlk, "A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem," Eur. J. Oper. Res., vol. 236, no. 3, pp. 800–810, Aug. 2014, doi: 10.1016/j.ejor.2013.08.032.
25. I. Elhallaoui, D. Villeneuve, F. Soumis, and G. Desaulniers, "Dynamic Aggregation of Set-Partitioning Constraints in Column Generation," Oper. Res., vol. 53, no. 4, pp. 632–645, Aug. 2005, doi: 10.1287/opre.1050.0222.
26. D. Guimarans, R. Herrero, D. Riera, A. A. Juan, and J. J. Ramos, "Combining probabilistic algorithms, Constraint Programming and Lagrangian Relaxation to solve the Vehicle Routing Problem," Ann. Math. Artif. Intell., vol. 62, no. 3–4, pp. 299–315, Jul. 2011, doi: 10.1007/s10472-011-9261-y.
27. C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems, vol. 5. Boston, MA: Springer US, 2002.
28. Z. Liu, K. Zhou, F. Jiang, and Y. Zhen, "Application of MTabu in VRPTW," Dec. 2017, doi: 10.1109/ICCSEC.2017.8446794.
29. H. Kurnia, E. G. Wahyuni, E. C. Pembrani, S. T. Gardini, and S. K. Aditya, "Vehicle Routing Problem Using Genetic Algorithm with Multi Compartment on Vegetable Distribution," IOP Conf. Ser. Mater. Sci. Eng., vol. 325, Mar. 2018, doi: 10.1088/1757-899X/325/1/012012.
30. I. Ramírez Cortes, J. A. Hernández Aguilar, M. Ángel Ruiz, M. A. Cruz-Chavez, and G. Arroyo-Figueroa, "Design and Implementation of a CVRP Simulator Using Genetic Algorithms," Res. Comput. Sci., vol. 147, no. 2, Dec. 2018, doi: 10.13053/rcs-147-2-3.
31. R. Fitriana, P. Moengin, and U. Kusumaningrum, "Improvement Route for Distribution Solutions MDVRP (Multi Depot Vehicle Routing Problem) using Genetic Algorithm," IOP Conf. Ser. Mater. Sci. Eng., vol. 528, Jun. 2019, doi: 10.1088/1757-899X/528/1/012042.
32. B. Rabbouch, F. Saâdaoui, and R. Mraihi, "Efficient implementation of the genetic algorithm to solve rich vehicle routing problems," Oper. Res., Oct. 2019, doi: 10.1007/s12351-019-00521-0.
33. J. Euchi and A. Sadok, "Hybrid genetic-sweep algorithm to solve the vehicle routing problem with drones," Phys. Commun., vol. 44, Feb. 2021, doi: 10.1016/j.phycom.2020.101236.
34. Mahrach, M., Miranda, G., León, C., & Segredo, E. (2020). Comparison between Single and Multi-Objective Evolutionary Algorithms to Solve the Knapsack Problem and the Travelling Salesman Problem. Mathematics, 8(11), 2018. https://doi.org/10.3390/math8112018
35. S. MNASRI, N. NASRI, A. VAN DEN BOSSCHE and T. VAL, "3D indoor redeployment in IoT collection networks: a real prototyping using a hybrid PI-NSGA-III-VF," 2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC), 2018, pp. 780-785, doi: 10.1109/IWCMC.2018.8450372.
36. Mnasri, S., Nasri, N., Alrashidi, M. et al. IoT networks 3D deployment using hybrid many-objective optimization algorithms. J Heuristics 26, 663–709 (2020). https://doi.org/10.1007/s10732-020-09445-x
37. T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, "A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms," Soft Comput., vol. 23, no. 9, pp. 3137–3166, May 2019, doi: 10.1007/s00500-017-2965-0.
38. Ngo Tung Son, Jafreezal Jaafar, Izzatdin Abdul Aziz, and Nguyen Hoang Giang. 2021. Introduction to A Compromise Programming Based Method for Complex Scheduling and Planning Problems. In Proceedings of the 2021 4th International Conference on Information Management & Management Science (IMMS '21). ACM, New York, NY, USA, 133-138. https://doi.org/10.1145/3485190.3485231.
39. J. L. Ringuest, "Compromise Programming," in Multiobjective Optimization: Behavioral and Computational Considerations, Boston, MA: Springer US, 1992.
40. S. Tung Ngo, J. Jafreezal, G. Hoang Nguyen, and A. Ngoc Bui, "A Genetic Algorithm for Multi-Objective Optimization in Complex Course Timetabling," in 2021 10th International Conference on Software and Computer Applications, Feb. 2021, pp. 229–237, doi: 10.1145/3457784.3457821.
41. S. T. Ngo, J. B. Jaafar, I. A. Aziz, G. H. Nguyen, and A. N. Bui, "Genetic Algorithm for Solving Multi-Objective Optimization in Examination Timetabling Problem," Int. J. Emerg. Technol. Learn., vol. 16, no. 11, p. 4, Jun. 2021, doi: 10.3991/ijet.v16i11.21017.
42. S. T. Ngo, J. Jaafar, I. A. Aziz, and B. N. Anh, "A Compromise Programming for Multi-Objective Task Assignment Problem," Computers, vol. 10, no. 2, Jan. 2021, doi: 10.3390/computers10020015.
43. T. S. Ngo et al., "Some Algorithms to Solve a Bi-Objectives Problem for Team Selection," Appl. Sci., vol. 10, no. 8, Apr. 2020, doi: 10.3390/app10082700.
44. N. T. Son, J. Jaafar, I. A. Aziz, and B. N. Anh, "Meta-Heuristic Algorithms for Learning Path Recommender at MOOC," IEEE Access, pp. 1–1, 2021, doi: 10.1109/ACCESS.2021.3072222.
45. Son, N. T., Jaafar, J., Aziz, I. A., Anh, B. N., Binh, H. D. et al. (2021). A Compromise Programming to Task Assignment Problem in Software Development Project. CMC-Computers, Materials & Continua, 69(3), 3429–3444.

46. Ngo Tung Son, Jafreezal Jaafar, Doan Van Thang, Phan Lac Duong, and Bui Ngoc Anh. 2021. The Effectiveness of Reference Point Selection Methods for Compromise Programming in Multi-Criteria Learning Path Search Algorithm. In Proceedings of the 2021 4th International Conference on Information Management & Management Science (IMMS '21). ACM, New York, NY, USA, 133-138. https://doi.org/10.1145/3485190.3485236.
47. Katoch, S., Chauhan, S.S. & Kumar, V. A review on genetic algorithm: past, present, and future. Multimed Tools Appl 80, 8091–8126 (2021). https://doi.org/10.1007/s11042-020-10139-6.
48. de Oca M.A.M., Cotta C., Neri F. (2012) Local Search. In: Neri F., Cotta C., Moscato P. (eds) Handbook of Memetic Algorithms. Studies in Computational Intelligence, vol 379. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-23247-3_3.
49. Gendreau M., Potvin JY. (2005) Tabu Search. In: Burke E.K., Kendall G. (eds) Search Methodologies. Springer, Boston, MA. https://doi.org/10.1007/0-387-28356-0_6.
50. Ishibuchi H., Imada R., Masuyama N., Nojima Y. (2019) Comparison of Hypervolume, IGD and IGD+ from the Viewpoint of Optimal Distributions of Solutions. In: Deb K. et al. (eds) Evolutionary Multi-Criterion Optimization. EMO 2019. Lecture Notes in Computer Science, vol 11411. Springer, Cham. https://doi.org/10.1007/978-3-030-12598-1_27
51. M. T. M. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: fundamentals and evolutionary methods," Nat. Comput., vol. 17, no. 3, pp. 585–609, Sep. 2018, doi: 10.1007/s11047-018-9685-y.