7-2021

An Adaptive Large Neighborhood Search for the green mixed fleet vehicle routing problem with realistic energy consumption and partial recharges

Vincent F. YU
National Taiwan University of Science and Technology

Panca JODIAWAN
National Taiwan University of Science and Technology

Aldy GUNAWAN
Singapore Management University, aldygunawan@smu.edu.sg

An Adaptive Large Neighborhood Search for the green mixed fleet vehicle routing problem with realistic energy consumption and partial recharges

Vincent F. Yu [a,b], Panca Jodiawan [a,*], Aldy Gunawan [c]

a Department of Industrial Management, National Taiwan University of Science and Technology, Taipei, Taiwan
b Center for Cyber–Physical System Innovation, National Taiwan University of Science and Technology, Taipei, Taiwan
c School of Computing and Information Systems, Singapore Management University, Singapore

Published in Applied Soft Computing, 105, July 2021, 107251. DOI: 10.1016/j.asoc.2021.107251

Abstract: This study addresses a variant of the Electric Vehicle Routing Problem with Mixed Fleet, named as the Green Mixed Fleet Vehicle Routing Problem with Realistic Energy Consumption and Partial Recharges. This problem contains three important characteristics — realistic energy consumption, partial recharging policy, and carbon emissions. An adaptive Large Neighborhood Search heuristic is developed for the problem. Experimental results show that the proposed ALNS finds optimal solutions for most small-scale benchmark instances in a significantly faster computational time compared to the performance of CPLEX solver. Moreover, it obtains high quality solutions for all medium- and large-scale instances under a reasonable computational time. We also perform numerical studies to analyze the potential carbon emission reduction resulted from the proposed model.

Keywords: Electric vehicle routing problem, Mixed fleet, Emission minimization, Adaptive large neighborhood search

Corresponding author: Panca Jodiawan, pancajodiawan@gmail.com

1. Introduction

Global warming has turned out to be a growing concern worldwide due to various human activities [1], with transportation systems becoming one of the largest contributors of air pollution for many countries; e.g., China [2], U.S.A. [3], and EU-27 [4]. Air pollution has been proven to toxicologically affect people's health [5], [6], as it not only causes negative impacts toward the environment, but also slows down global economic growth [7]. Considering such effects, innovations to decrease pollution are greatly needed. In addition, governments of various nations have recently set targets and strategies for emissions reduction; i.e. the U.S. proposed a target on greenhouse gas (GHG) emissions reduction to be 26%–28% below 2005 levels by 2025 [8]; and the EU-27 has also set a similar goal to reduce GHG emissions to be 60% below 1990 levels by 2050 [9].

The aforementioned actions have led to increasing attention toward the utilization of electric vehicles (EVs). The main benefit of EVs compared to conventional vehicles, denoted as internal combustion vehicles (ICVs), is that EVs produce zero tailpipe emission when operating [10], although several considerations need to be carefully studied [11], [12]. Various logistics companies, e.g. DHL, UPS, and FedEx [13], [14], [15], have begun to utilize EVs as part of their operational fleets for conducting deliveries. In spite of the sustainability-related potential benefit offered by harnessing EVs, several limitations co-exist, which impede their adoption level; e.g., high purchase prices and battery costs, limited driving range, and potentially long recharging time requirements [16]. Considering the benefits and limitations of utilizing EVs, careful planning needs to be conducted by companies before any mass adoption of EVs.

One of the main activities carried out in a logistics company is distributing goods by visiting a set of customers with a set of available vehicles that start and end at the depot and minimizing transportation cost, which is generally called a vehicle routing problem (VRP) [17]. Due to VRP's important economic applications and theoretical interests, the related research field has grown fast and intensive over the years by considering various real-world constraints and challenges [18]. One of the most recent practical situations considered in VRP involves the utilization of EVs. Schneider et al. [19] introduced Electric Vehicle Routing Problem with Time Windows (EVRPTW) by incorporating the recharging possibility at any of the available stations with recharging times that depend on the battery level when arriving at the station and by considering capacity constraints on vehicles as well as customer time windows. The problem is formulated as a mixed-integer programming model. A hybrid Variable Neighborhood Search with Tabu Search (VNS/TS) heuristic, in which infeasible solutions are allowed during the search, was proposed

to solve the problem. The performance of the algorithm was tested with new generated instances. Furthermore, the authors made a comparison to the results obtained by VNTS/TS without Simulated Annealing (SA) and showed that SA acceptance criterion performed better versus simply accepting an improved solution criterion.

Research in dealing with EVs in the VRP field has also started to gain momentum. Keskin and Çatay [20] extended EVRPTW by considering a partial recharge policy and proposing Adaptive Large Neighborhood Search to solve the problem. They proved that a 1.64% cost saving can be obtained by applying the partial recharge policy based on a comparison with the result of Schneider et al. [19]. The research was later extended by including the existence of multiple recharging technologies [21]. Felipe et al. [22] pioneered the research on multiple refueling technology, although they did not explicitly mention EVs. Keskin and Çatay [23] dealt with multiple recharging technologies for EVRPTW and found several benefits: lower total energy consumption and lower number of utilized fleets. Another characteristic worth mentioning in the literature is the recharging time characteristic. The real recharging function is generally non-linear with time [24]. Zündorf [25] targeted the non-linear recharging function by transforming the non-linearity into a piecewise linear function. Montoya et al. [26] introduced the electric vehicle routing problem with a non-linear recharging process by utilizing the piecewise linear approximation introduced by Zündorf [25]. In addition to electric vehicles, hybrid vehicles which utilize both fuel and electricity have also been identified to have such potential benefits [27].

Electric vehicle employment in city logistics, which involve a two-tier supply chain network, has gained greater interest. Breunig et al. [28] considered EVs as part of the operational fleets in 2EVRP and later termed this as E2EVRP. In the problem, a fleet of ICVs still serves the first echelon, while EVs operate on the second echelon. Jie et al. [29] also worked on a similar problem with the main difference being that the considered type of EVs requires battery swapping instead of recharging.

A mixed fleet between EVs and ICVs has recently become a prominent strategy due to the limitations encountered in EVs' utilization. Goeke and Schneider [30] addressed EVRPTW-MF, involving a realistic energy consumption model, to calculate electricity consumption of an EV and fuel consumption of an ICV. In order to manage the driving range limitation of EVs, it is important to adopt a realistic energy consumption model to predict electricity usage [31]. Hiermann et al. [32] extended the mixed fleet scenario by considering Plug-in Hybrid Electric Vehicles (PHEVs) together with EVs and ICVs. Two types of sensitivity analyses were performed to analyze the benefit of utilizing a mixed fleet compared to a single type of particular fleet as well as the impact of cost change toward the fleet composition. Macrina et al. [33] proposed the green mixed fleet vehicle routing problem with partial recharging and time windows, involving a limitation of carbon emissions that can be emitted by a company's operational fleet. Moreover, Macrina et al. [34] dealt with a mixed fleet scenario of EVs and ICVs by considering partial recharges and a realistic energy consumption model.

Based on the aforementioned literature, the mixed fleet problem seems to be widely adopted by logistics companies due to current market and business conditions. Recently, Yu et al. [35] proposed a new variant of a mixed fleet between EVs and ICVs, named as the Green Mixed Fleet Vehicle Routing Problem with Realistic Energy Consumption and Partial Recharges (GMFVRP-REC-PR) and developed a mixed integer programming model for the problem. It simultaneously addresses three important characteristics, partial recharges, realistic energy consumption, and emissions, making the problem closer to the real-world condition. The problem is modeled as a single objective problem with economic and environmental considerations, i.e. transportation and carbon emissions costs, respectively. This modeling approach has also been employed in various routing problem variants [36–38]. Based on the findings presented in Yu et al. [35], the computational time of solving the mathematical model is significantly high, even for the small-scale instances. Therefore, this research is presented with following objectives and contributions:

1. To propose an Adaptive Large Neighborhood Search (ALNS) embedded with a dynamic programming procedure and an efficient solution evaluation procedure to solve GMFVRP-REC-PR of various sizes. In addition, both feasible and infeasible solution spaces are considered in aim of guiding the search toward diverse regions.
2. To analyze the potential carbon emission reduction of the proposed GMFVRP-REC-PR.

The remainder of the paper is organized as follows. Section 2 describes GMFVRP-REC-PR and explains our MILP formulation. Section 3 proposes the ALNS algorithm in detail. Section 4 reports computational studies involving the parameter settings and makes a comparative analysis between our proposed methods with state-of-the-art benchmark results and carbon emission reduction analysis. Finally, Section 5 presents our conclusions and future direction of this research.

2. Model formulation

This section explains the mathematical model of GMFVRP-REC-PR formulated by Yu et al. [35]. Section 2.1 presents the energy consumption model, which will be used to calculate an EV's electricity consumption and an ICV's fuel consumption. Section 2.2 explains the mathematical model of GMFVRP-REC-PR in detail. In addition, all the variables used in the mathematical model and equations are summarized in Table A.1 of the Appendix.

2.1. Electricity and fuel consumption calculations

In order to obtain an EV's energy consumption, a mechanical power requirement P_M is required. We evaluate P_M by using the energy consumption model presented in Bektaş and Laporte [39], where P_M depends on several variables: mass, speed, gradient, and physical environment (road surface, vehicle dimensions, and engine properties). Eq. (1) calculates the amount of P_M.

$$P_M = \left(m.a + \frac{1}{2}.c_d.\rho.A.v^2 + m.g.\sin(\alpha) + c_r.m.g.\cos(\alpha) \right).v \quad (1)$$

where m is vehicle mass (kg), a is acceleration (m/s^2), c_d is the coefficient of aerodynamic drag, ρ is air density (kg/m^3), A is vehicle frontal surface (m^2), v is vehicle speed, g is a gravitational constant (m/s^2), α is the gradient angle of the road, and c_r is the coefficient of rolling resistance.

We particularly define m as a function of the currently loaded amount of cargo that is brought by vehicle k from a particular node i to node j, u_{ij}^k. Therefore, $m\left(u_{ij}^k\right)$ represents the total mass consisting of a vehicle's curb mass and u_{ij}^k. Let $p_{ij}^k\left(u_{ij}^k\right)$ denote the constant mechanical power requirement for an EV to travel from node i to node j with total mass $m\left(u_{ij}^k\right)$. Thus, $p_{ij}^k\left(u_{ij}^k\right)$ is able to be evaluated by Eq. (2).

$$p_{ij}^k\left(u_{ij}^k\right) = \left(\frac{1}{2}.c_d.\rho.A.v^2 + m\left(u_{ij}^k\right).g.(\sin\left(\alpha_{ij}\right) + c_r.\cos\left(\alpha_{ij}\right)) \right).v_{ij} \quad (2)$$

The electricity consumption of an EV can be obtained from the calculated mechanical power requirement by the following two steps. First, the necessary electric power is evaluated by taking energy losses that occur in the electric engine into account by multiplying $p_{ij}^{k}\left(u_{ij}^{k}\right)$ with energy efficiency, Φ^{d} and discharging efficiency, φ^{d}. Second, the final electricity requirement is obtained by considering the travel time from node i to node j, t_{ij}. To sum up, Eq. (3) calculates the electricity consumption of an EV while traveling from node i to node j.

$$b_{ij}^{k}\left(u_{ij}^{k}\right)=\Phi^{d}.\varphi^{d}.p_{ij}^{k}\left(u_{ij}^{k}\right).t_{ij} \tag{3}$$

The aforementioned mechanical power is also utilized to evaluate the fuel consumption of an ICV. First, the fuel consumption rate of ICV k while traveling from node i to j is calculated by:

$$FR_{ij}^{k}\left(u_{ij}^{k}\right)=\max\left(\frac{\xi}{\kappa.\psi}\left(k.N.D+\frac{p_{ij}^{k}\left(u_{ij}^{k}\right)}{\eta.\eta_{tf}}\right),0\right) \tag{4}$$

where ξ denotes the fuel-to-air mass ratio, κ represents the heating value of typical diesel fuel, k represents the engine friction factor, N denotes engine speed, D represents engine displacement, ψ denotes a factor converting the fuel rate from gram per second to liters per second, η represents the efficiency parameter for diesel engines, and η_{tf} denotes drive train efficiency. After obtaining the fuel consumption rate, the associated fuel consumption of an ICV traveling from node i to node j can be calculated by:

$$f_{ij}^{k}\left(u_{ij}^{k}\right)=FR_{ij}^{k}\left(u_{ij}^{k}\right).t_{ij} \tag{5}$$

2.2. Problem definition

Consider a directed graph $G=(0\cup V\cup F'\cup N+1,A)$, where V is a set of N customers, $V=\{1,\ldots,N\}$, F is a set of recharging stations, F‘ is the set of dummy nodes representing each node in F to enable multiple visits to a particular recharging station in the formulation, and nodes 0 and $N+1$ are the starting and ending points, respectively. The nodes are connected through a set of arcs $A=\{(i,j)|i,j\in 0\cup V\cup F'\cup N+1, i\neq j\}$ with a non-negative distance, d_{ij}, travel time t_{ij}, and cost c_t for each kilometer traveled by the vehicle. For simplifying the mathematical model formulation, we introduce several sets: $V'=V\cup F'$, $V_0=V\cup 0$, $V_{N+1}=V\cup N+1$, $V_0'=V\cup F'\cup 0$, and $V_{N+1}'=V\cup F'\cup N+1$.

Each node i has non-negative demand, $q_i(i\in V)$, service time, $s_i(i\in V)$, and hard time windows, $[e_i,l_i](i\in 0\cup V\cup F'\cup N+1)$, where early arrival with waiting is allowed, but late arrival is strictly prohibited. Sets of EVs, K_E, and ICVs, K_{IC}, with capacity Q are available at the depot. EVs having a battery capacity of B can be charged partially at a recharging rate of r.

The carbon emissions emitted by an ICV depend on the CO_2 emitted per liter of fuel, FE [33] which is later multiplied with the cost c_e to obtain the carbon emission cost. Let EM_{ij} represent the total carbon emissions produced by an ICV while traveling from node i to node j, calculated by:

$$EM_{ij}=FE.FR_{ij} \tag{6}$$

where FE is a fuel conversion factor, expressed in grams of CO_2 emitted per liter of fuel. The value of FE depends on several factors: vehicle type, fuel type, etc. FR_{ij} is the fuel consumption of an ICV, as has been explained in Section 2.1.

Decision variables

- $x_{ijk_E}^{E}=1$ if an EV k_E travels from node i to j $(k_E\in K_E; i,j\in 0\cup V\cup F'\cup N+1)$
- $x_{ijk_{IC}}^{JC}=1$ if an ICV k_{IC} travels from node i to j $(k_{IC}\in K_{IC}; i,j\in 0\cup V\cup N+1)$
- $\tau_i=$ arrival time at node $i(i\in V_0')$
- $\tau_{N+1}^{k_E}=$ arrival time of an EV k_E at node $N+1(k_E\in K_E)$
- $\tau_{N+1}^{k_{IC}}=$ arrival time of an ICV k_{IC} at node $N+1(k_{IC}\in K_{IC})$
- $u_0^{k_E}=$ initial load brought by an EV $k_E(k_E\in K_E)$
- $u_0^{k_{IC}}=$ initial load brought by an ICV $k_{IC}(k_{IC}\in K_{IC})$
- $u_{ij}^{k_{IC}}=$ amount of load brought by an ICV k_{IC} when traveling from node i to node j $(k_{IC}\in K_{IC}; i,j\in 0\cup V\cup N+1)$
- $u_{ij}^{k_E}=$ amount of load brought by an ICV k_E when traveling from node i to node j $(k_E\in K_E; i,j\in 0\cup V\cup F'\cup N+1)$
- $y_i^{K_E}=$ remaining electric energy of an EV k_E upon arrival at node i $(k_E\in K_E; i,j\in 0\cup V\cup F'\cup N+1)$
- $Y_i^{K_E}=$ amount of electric energy obtained by an EV k_E after recharging at recharging station i $(k_E\in K_E; i\in F)$
- $p_{ij}^{k_E}\left(u_{ij}^{k_E}\right)=$ amount of mechanical power spent by an EV k_E when traveling from node i to node j and carrying a load of $u_{ij}^{k_E}(k_E\in K_E; i,j\in 0\cup V\cup F'\cup N+1)$
- $p_{ij}^{k_{IC}}\left(u_{ij}^{k_{IC}}\right)=$ amount of mechanical power spent by an ICV k_{IC} when traveling from node i to node j and carrying a load of $u_{ij}^{k_{IC}}(k_{IC}\in K_{IC}; i,j\in 0\cup V\cup N+1)$
- $b_{ij}^{k_E}\left(u_{ij}^{k_E}\right)=$ amount of electric energy consumed by an EV k_E when traveling from node i to node j and carrying a load of $u_{ij}^{k_E}(k_E\in K_E; i,j\in 0\cup V\cup F'\cup N+1)$
- $f_{ij}^{k_{IC}}\left(u_{ij}^{k_{IC}}\right)=$ amount of fuel consumed by an ICVk_{IC} when traveling from node i to node j and carrying a load of $u_{ij}^{k_{IC}}(k_{IC}\in K_{IC}; i,j\in 0\cup V\cup N+1)$

Objective Function

$$\begin{aligned}\min &\sum_{k_E\in K_E}\sum_{i\in V_0'}\sum_{j\in V_{N+1}'}c_t d_{ij}x_{ijk_E}^{E}+\sum_{k_{IC}\in K_{IC}}\sum_{i\in V_0'}\sum_{j\in V_{N+1}'}c_t d_{ij}x_{ijk_{IC}}^{JC}\\&+\sum_{k_{IC}\in K_{IC}}\sum_{i\in V_0'}\sum_{j\in V_{N+1}'}c_e FE f_{ij}^{k_{IC}}(u_{IC})\end{aligned} \tag{7}$$

Constraints

$$\sum_{k_E\in K_E}\sum_{j\in V_{N+1}'}x_{ijk_E}^{E}+\sum_{k_{IC}\in K_{IC}}\sum_{j\in V_{N+1}}x_{ijk_{IC}}^{JC}=1\quad\forall i\in V \tag{8}$$

$$\sum_{j\in V_{N+1}'}x_{ijk_E}^{E}\leq 1\ \ \forall i\in V_0',\forall k_E\in K_E \tag{9}$$

$$\sum_{j\in V_{N+1}}x_{ijk_{IC}}^{IC}\leq 1\ \ \forall i\in V_0,\forall k_{IC}\in K_{IC} \tag{10}$$

$$\sum_{i\in V_0'}x_{ijk_E}^{E}=\sum_{i\in V_{N+1}'}x_{jik_E}^{E}\ \ \forall j\in V',\forall k_E\in K_E \tag{11}$$

$$\sum_{i\in V_0}x_{ijk_{IC}}^{JC}=\sum_{i\in V_{N+1}}x_{jik_{IC}}^{JC}\ \ \forall j\in V,\forall k_{IC}\in K_{IC} \tag{12}$$

$$\sum_{j\in V'}x_{0jk_E}^{E}\leq K_E\ \ \forall k_E\in K_E \tag{13}$$

$$\sum_{j\in V}x_{0jk_{IC}}^{JC}\leq K_{IC}\ \ \forall k_{IC}\in K_{IC} \tag{14}$$

$$\tau_0=0 \tag{15}$$

$$\begin{aligned}&\tau_i+(s_i+t_{ij})\left(\sum_{k_E\in K_E}x_{ijk_E}^{E}+\sum_{k_{IC}\in K_{IC}}x_{ijk_{IC}}^{JC}\right)\\&-M\left(1-\left(\sum_{k_E\in K_E}x_{ijk_E}^{E}+\sum_{k_{IC}\in K_{IC}}x_{ijk_{IC}}^{JC}\right)\right)\leq\tau_j\end{aligned}\quad\forall i\in V_0,\forall j\in V \tag{16}$$

$$\tau_i+(s_i+t_{ij})\sum_{k_E\in K_E}x_{ijk_E}^{E}-M\left(1-\sum_{k_E\in K_E}x_{ijk_E}^{E}\right)\leq\tau_j\ \ \forall i\in V_0,\forall j\in F' \tag{17}$$

$$\tau_i + \frac{1}{r}\left(Y_i^{k_E} - y_i^{k_E}\right) + t_{ij}x_{ijk_E}^{E} - M(1 - x_{ijk_E}^{E}) \le \tau_j \ \ \forall i \in F', \forall j \in V', \forall k_E \in K_E \tag{18}$$

$$\tau_i + \frac{1}{r}\left(Y_i^{k_E} - y_i^{k_E}\right) + t_{iN+1}x_{iN+1k_E}^{E} - M(1 - x_{iN+1k_E}^{E}) \le \tau_{N+1}^{k_E} \ \forall i \in F', \forall k_E \in K_E \tag{19}$$

$$\tau_i + (s_i + t_{iN+1})x_{iN+1k_E}^{E} - M\left(1 - x_{iN+1k_E}^{E}\right) \le \tau_{N+1}^{k_E} \ \ \forall i \in V_0, \forall k_E \in K_E \tag{20}$$

$$\tau_i + (s_i + t_{iN+1})x_{iN+1k_{IC}}^{IC} - M\left(1 - x_{iN+1k_{IC}}^{IC}\right) \le \tau_{N+1}^{k_{IC}} \ \ \forall i \in V_0, \forall k_{IC} \in K_{IC} \tag{21}$$

$$e_i \le \tau_i \le l_i \ \ \forall i \in V' \tag{22}$$

$$e_{N+1} \le \tau_{N+1}^{k_{IC}} \le l_{N+1} \tag{23}$$

$$e_{N+1} \le \tau_{N+1}^{k_E} \le l_{N+1} \tag{24}$$

$$\sum_{i \in V_0'} u_{ij}^{k_E} - \sum_{i \in V_{N+1}'} u_{ji}^{k_E} = q_j \sum_{i \in V_0'} x_{ijk_E}^{E} \ \ \forall j \in V', \forall k_E \in K_E \tag{25}$$

$$\sum_{i \in V'} u_{iN+1}^{k_E} = 0 \ \ \forall k_E \in K_E \tag{26}$$

$$0 \ \le u_{ij}^{k_E} \le Q.x_{ijk_E}^{E} \quad \forall i \in V_0', \forall j \in V_{N+1}', \forall k_E \in K_E \tag{27}$$

$$\sum_{i \in V_0} u_{ij}^{k_{IC}} - \sum_{i \in V_{N+1}} u_{ji}^{k_{IC}} = q_j \sum_{i \in V_0} x_{ijk_{IC}}^{IC} \ \ \forall j \in V, \forall k_{IC} \in K_{IC} \tag{28}$$

$$\sum_{i \in V} u_{iN+1}^{k_{IC}} = 0 \ \ \forall k_{IC} \in K_{IC} \tag{29}$$

$$0 \ \le u_{ij}^{k_{IC}} \le Q.x_{ijk_{IC}}^{IC} \quad \forall i \in V_0, \forall j \in V_{N+1}, \forall k_{IC} \in K_{IC} \tag{30}$$

$$p_{ij}^{k_{IC}}(u_{ij}^{k_{IC}}) = \left(\tfrac{1}{2}.c_d.\rho.A.v^2 + m_{truck}.g.c_r\right)x_{ijk_{IC}}^{IC} \quad \forall i \in V_0, \forall j \in V_{N+1}, \forall k_{IC} \in K_{IC} \\ +g.c_r.u_{ij}^{k_{IC}} \tag{31}$$

$$f_{ij}^{k_{IC}}(u_{ij}^{k_{IC}}) = \left(\frac{\xi}{\kappa.\psi}\left(kND + \frac{p_{ij}^{k_{IC}}\left(u_{ij}^{k_{IC}}\right)}{\eta.\eta_{tf}}\right)\right)x_{ijk_{IC}}^{IC}.t_{ij} \ \forall i \in V_0, \forall j \in V_{N+1}, \forall k_{IC} \in K_{IC} \tag{32}$$

$$p_{ij}^{k_E}(u_{ij}^{k_E}) = \left(\tfrac{1}{2}.c_d.\rho.A.v^2 + m_{truck}.g.c_r\right)x_{ijk_E}^{E} \quad \forall i \in V_0', \forall j \in V_{N+1}', \forall k_E \in K_E \\ +g.c_r.u_{ij}^{k_E} \tag{33}$$

$$b_{ij}^{k_E}(u_{ij}^{k_E}) \ge \Phi_d.\varphi_d.p_{ij}^{k_E}(u_{ij}^{k_E}).t_{ij} \ \ \forall i \in V_0', \forall j \in V_{N+1}', \forall k_E \in K_E \tag{34}$$

$$b_{ij}^{k_E}(u_{ij}^{k_E}) + M\left(1 - x_{ijk_E}^{E}\right) \le y_i^{k_E} - y_j^{k_E} \\ \le b_{ij}^{k_E}(u_{ij}^{k_E}) - M\left(1 - x_{ijk_E}^{E}\right) \quad \forall i \in V_0, \forall j \in V_{N+1}', \forall k_E \in K_E \tag{35}$$

$$b_{ij}^{k_E}(u_{ij}^{k_E}) + M\left(1 - x_{ijk_E}^{E}\right) \le Y_i^{k_E} - y_i^{k_E} \\ \le b_{ij}^{k_E}(u_{ij}^{k_E}) - M\left(1 - x_{ijk_E}^{E}\right) \quad \forall i \in F', \forall j \in V_{N+1}', \forall k_E \in K_E \tag{36}$$

$$y_0^{k_E} = B \tag{37}$$

$$y_i^{k_E} \le B \ \ \forall i \in V_0', \forall k_E \in K_E \tag{38}$$

$$y_i^{k_E} \le Y_i^{k_E} \ \ \forall i \in F', \forall k_E \in K_E \tag{39}$$

$$Y_i^{k_E} \le B \ \ \forall i \in F', \forall k_E \in K_E \tag{40}$$

Objective function (7) is to minimize the total cost comprised of the distance traveled cost by all utilized fleets and the total emitted pollution cost from the utilized fleet of ICVs. Eq. (8) ensures that each customer is visited at most once, either by an EV or an ICV. Eqs. (9) and (10) guarantee that each vehicle, either an EV or an ICV, can only visit one customer when it starts a route from the depot. Eqs. (11) and (12) are the flow-in flow-out constraints for an EV and an ICV, respectively. Eqs. (13) and (14) consecutively limit the number of EVs and ICVs that can be assigned to serve customers. Constraints that regulate time are described from Eq. (15) to Eq. (24). Eq. (15) guarantees that the initial time of every utilized fleet starts from 0. Eqs. (16) to (21) link arrival times at node i and node j and M denotes a very large value which is multiplied with the term(s) inside the parentheses. Eqs. (22) to (24) ensure that the time windows of each visited node are not violated. Eqs. (25) to (30) regulate the load balance. The total load carried by an EV and an ICV from a node to another node is modeled by Eqs. (25) and (28), respectively. Eqs. (26) and (29) guarantee that a vehicle returning to the depot carries no remaining load. Eqs. (27) and (30) respectively ensure that the total load brought by an EV or an ICV is within the maximum allowable load. Eqs. (31) and (32) calculate the fuel consumption of an ICV that travels through an arc (i, j). Eqs. (33) and (34) calculate the electricity consumption of an EV that travels through an arc (i, j). Eq. (35) is responsible for tracking the energy level of an EV traveling from node i to node j. Eq. (36) links the energy level when an EV performs a recharging process at a recharging station i. Eqs. (37) to (40) guarantee the natural range of an EV's battery level.

3. Proposed adaptive large neighborhood search algorithm for GMFVRP-REC-PR

Adaptive Large Neighborhood Search (ALNS), first proposed by Ropke and Pisinger [40], consists of removal and insertion heuristics. These heuristics are developed based on Large Neighborhood Search (LNS). ALNS differs from its previous version, LNS, as it utilizes more than one removal and more than one insertion heuristics, a removal and insertion heuristics selection mechanism that is based on collected statistics, and a simulated annealing metaheuristic for the acceptance mechanism. Several studies have showed the excellent performance of ALNS in solving various real-world transportation problems, like the orienteering problem [41] and vehicle routing problems [42–45]. ALNS has also been adopted to solve routing problems that consider electric vehicles' utilization [20,30,46].

This study embeds several mechanisms that enable ALNS to solve GMFVRP-REC-PR. First, the preprocessing stage is implemented to reduce the searching space by eliminating infeasible arcs. Second, infeasible solutions are allowed during the iterations. Therefore, this work utilizes a generalized objective function comprised of the objective function presented in Eq. (7) and several penalty values. Third, dynamic programming is integrated into ALNS to determine recharging station visit(s) for the routes of an EV in an optimal manner. Lastly, the time-efficient evaluation of neighborhoods is employed in this work to reduce the computational burden of implementing removal and insertion heuristics repeatedly.

The flow chart of the proposed ALNS is shown in Fig. 1. Before ALNS is implemented, the algorithm starts by finding an initial solution (Section 3.1), setting the necessary parameters, and set the initial solution to be the best solution σ^*. There are two types of termination criterion utilized by the proposed ALNS, either by maximum number of iterations, $\eta^{\max}$, or by maximum number of non-improvement iterations, $\eta_{noi}^{\max}$. First, ALNS will determine how many customers need to be removed, δ, from the current solution. This mechanism depends on three parameters: minimum number of selected customers, $\Omega_{\min}$, maximum number of selected customers, $\Omega_{\max}$, and collected scores, π^c. A neighborhood solution, σ', is then built by applying removal and insertion heuristics to the current solution, σ. Several necessary components for applying this procedure are (1) δ, (2) available removal heuristics, D , (3) available insertion heuristics, and R (4) collected scores of removal and insertion heuristics, π.

The local search procedure – *localSearch* (.) which is explained in Section 3.7 – will be implemented if the objective value of a neighborhood solution, $\lambda\left(\sigma'\right)$, is smaller than $\lambda\left(\sigma^*\right)\left(1+\delta'\right)$ where σ^* is the best solution so far. If the objective of the solution after implementing *localSearch* (.) is less than $\lambda\left(\sigma^*\right)\left(1+\delta^d\right)$,

then *dynamicProgramming* (.), described in Section 3.8, is executed.

The implementation of a simulated annealing framework in ALNS is explained as follows. It starts by calculating Δ, the difference between $\lambda(\sigma')$ and $\lambda(\sigma)$ which is obtained by subtracting $\lambda(\sigma)$ from $\lambda(\sigma')$. If Δ is negative, then it means σ' is better than σ regardless of its feasibility. This σ' is accepted as a new σ. Furthermore, if $\lambda(\sigma')$ is below $\lambda(\sigma^*)$, then σ' is accepted as a new σ^*. Another solution, σ_f', is generated by *generateFeasibleSolution* (.), which will be further described in Section 3.9. If the resulting σ_f' is feasible through the feasibility checking, *feasible*(.), and the objective value of σ_f' is below the objective value of the best-feasible solution so far σ_f^*, then σ_f^* and t_{imp} representing the iteration when the last improvements are updated. If Δ is positive, then σ' will still replace σ if a particular value obtained by generating a random number with a range between 0 and 1 is lower than $e^{\frac{-\Delta}{T}}$, where T is current temperature, a parameter of simulated annealing. The detail calculation of Δ is further explained in Section 3.11. Moreover, a restarting strategy by adopting σ_f^* to be σ will be utilized every time a particular number of iterations, η^{res}, is reached.

The update score procedure – *updateScores*(.) – becomes the next step after performing the simulated annealing acceptance mechanism. This procedure which will be explained in Section 3.10 is defined as a function to update the scores of each removal and insertion heuristics. The scores of these heuristics later determine the tendency of each heuristic being selected during iterations, denoting a selection probability of a heuristic. The selection probability will be updated for each η^p iterations, and the penalty value of the generalized objective function will also be updated every time η^s iterations have been reached. Before ending a particular iteration, T decreases to αT.

3.1. Initial solution

Before generating an initial solution, a pre-processing step is performed to remove all arcs that clearly cannot be part of a feasible solution; i.e., their inclusion leads to constraint violation(s). The pre-processing steps are mentioned as follows.

1. $v, w \in V \wedge q_v + q_w > 0$
2. $v \in V_0', w \in V_{N+1}' \wedge e_v + s_v + t_{vw} > l_w$
3. $v \in V_0', w \in V' \wedge e_v + s_v + t_{vw} + s_w + t_{wN+1} > l_0$
4. $v \in V_0', w \in V' \wedge b_{vw}(q_w) > B$
5. $v \in V_0', w \in V' \wedge \forall j \in F_0', i \in F_{N+1}' : b_{jv}(q_v + q_w) + b_{vw}(q_w) + b_{wi}(0) > B$

Rules 1–3 eliminate arcs that violate the capacity and time windows restriction. These rules were initially utilized by Schneider et al. [19]. Rules 4 and 5 were specifically developed by Goeke and Schneider [30] and only hold for EV routes. Rule 4 judges those arcs that cannot be traveled even by an EV that has full battery capacity and only carries demand q_w along the arc (v, w). Rule 5 determines whether an arc is feasible or not by checking the total utilized electricity if the EV comes from a recharging station node or depot prior to traveling the arc (v, w) and visiting a recharging station node or depot after traveling the arc (v, w). If the total utilized electricity is larger than battery capacity in all scenarios, then the arc could be deemed infeasible.

After the preprocessing process has been done, an initial solution is created. All the customers are first placed on a removal list. Regret-two insertion heuristic – described in Section 3.6 – is then performed to put all the customers into the route based on a generalized objective function. After all the customers have been assigned to either an EV or an ICV, a local search procedure is then performed. Lastly, dynamic programming is applied to further improve the initial solution by optimally solving the recharging station visits in every route of employed EVs.

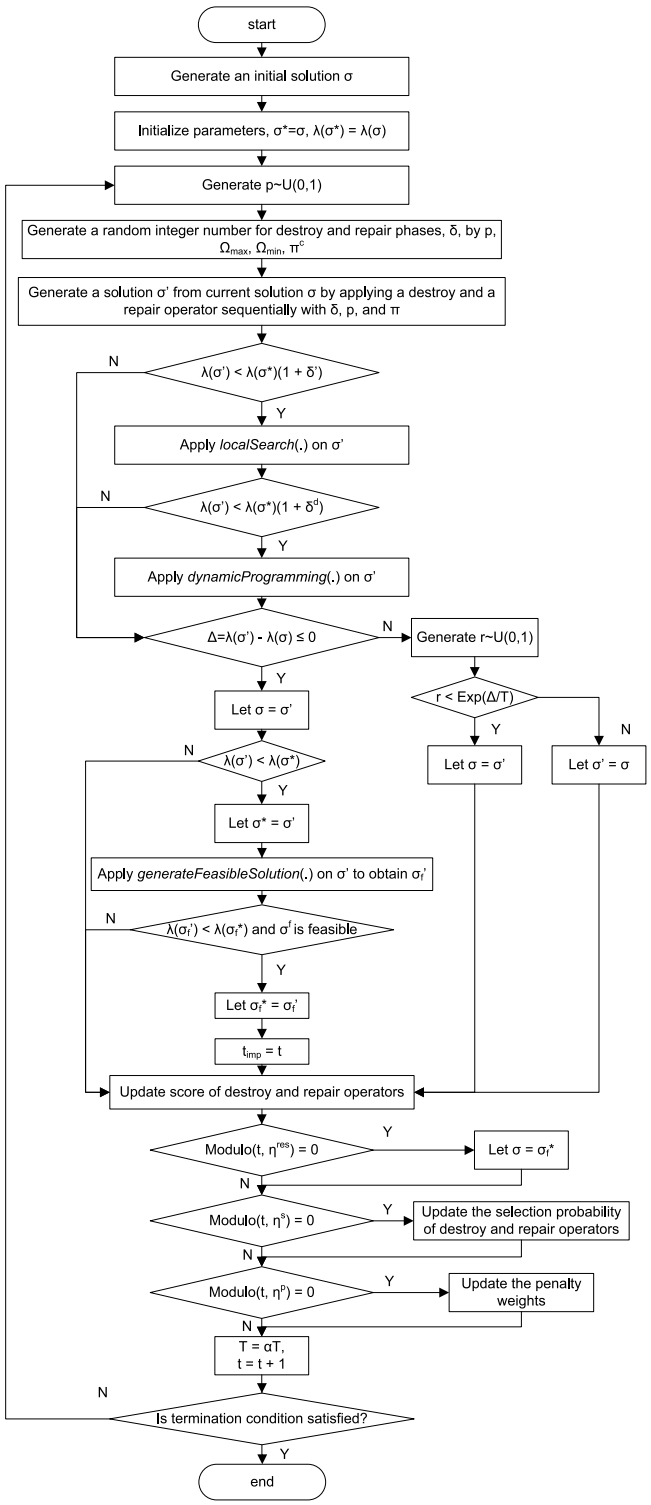


Fig. 1. Flow chart of the proposed ALNS for GMFVRP-REC-PR.

3.2. Generalized cost function

Infeasible solutions are allowed during the search in order to widen the solution spaces that can be traversed. Therefore, the generalized objective function of a solution, $\lambda_{gen}(\sigma)$, is calculated

by the following equation.

$$\lambda_{gen}(\sigma) = \lambda(\sigma) + \gamma_{cap}.Cap(\sigma) + \gamma_{tw}.TW(\sigma) + \gamma_{batt}.BT(\sigma) \quad (41)$$

Here, $\lambda_{gen}(\sigma)$ denotes the original objective function (7). There are three conditions where penalties apply: violations on capacity, time windows, and battery capacity, respectively denoted by γ_{cap}, γ_{tw}, and γ_{batt}.

The penalty factors are dynamically adjusted in the following manner. If there is at least one violation that occurred in η^p iterations, then the penalty multipliers are multiplied by a factor Ω in order to increase the possibility of finding feasible solutions; otherwise, the penalty multipliers are then divided by Ω.

3.3. Concatenation operators

A consideration of infeasible solutions means that the algorithm needs to calculate the total violation every time a new neighborhood solution is generated. Thus, efficiently evaluating a new neighborhood is important for the proposed ALNS algorithm in order to minimize the computational time. Nagata et al. [47] pioneered the development of an efficient approach that takes $O(1)$ time to evaluate a time window violation and to implement it in order to solve the vehicle routing problem with time windows. Schneider et al. [19] extended the approach that also takes $O(1)$ time to evaluate a battery penalty to deal with EVRPTW. In this work, we utilize the approach proposed by Schiffer and Walther [46] — namely, the corridor-based approach due to its ability to deal with a partial recharging policy. Forward and backward penalties of time window (TW) and battery (BT), represented respectively by $\vec{a}$ and $\leftarrow a$ where $a \in \{TW, BT\}$, are the required information to evaluate neighborhood solutions efficiently. This approach is particularly able to compute the time window and battery penalties in $O(1)$ time if the case is that one node is inserted between two partial routes. To evaluate the case in which two or more nodes are inserted, either forward or backward penalties need to be extended prior to the evaluation.

Several variables are needed to estimate the time window and battery forward and backward penalties. Table 1, which is also described in Schiffer and Walter (2017), summarizes the variables and their definitions for calculating the forward penalties, while the variables for calculating the backward penalties are defined as b^k with $k \in \{\min, \max, sl, rt, add\}$, $\tilde{b}_i^{\min}$, and $\tilde{b}_i^{\max}$. Readers may to refer to Schiffer and Walther [46] for more details in calculating each variable and penalty. One slight modification in the part of calculating electricity consumption is implemented, because this work considers a realistic energy consumption model. Hence, the electricity consumption of an EV while traveling from vertex i to vertex j is calculated as $h_{ij} = r.b_{ij}^{k_E}\left(u_{ij}^{k_E}\right)$, where $b_{ij}^{k_E}\left(u_{ij}^{k_E}\right)$ is obtained by Eq. (3).

The route concatenations are then explained as follows. The first scenario is insertion of a vertex into two partial routes, and the latter is the concatenation of two partial routes. Let a route $\sigma_i = \{0, \ldots, x, v, y, \ldots, n+1\}$, which is built by inserting vertex u between a partial route $\sigma_1 = \{0, \ldots, x\}$ and $\sigma_2 = \{y, \ldots, n+1\}$. The time window penalty of σ can be calculated by Eq. (42). Subsequently, the battery penalty of σ is calculated by utilizing Eq. (43).

$$\begin{aligned} TW(\sigma_i) = {} & \overrightarrow{TW}(\sigma_1) + \overleftarrow{TW}(\sigma_2) + \max\left\{0, a_v^{\min} - l_v - \max\left\{0, a_v^{\min} - a_v^{\max}\right\}\right\} \\ & + \max\left\{0, \min\left\{l_v, \max\left\{e_v, a_v^{\min}\right\}\right\}\right\} \\ & - b_v^{\min} - \max\left\{b_v^{\max} - b_v^{\min}, 0\right\}\right\} \end{aligned} \quad (42)$$

$$BT(\sigma_i) = \overrightarrow{BT}(\sigma_1) + \overleftarrow{BT}(\sigma_2) + \max\left\{0, a_v^{\min} - a_v^{\max}\right\} + D \quad (43)$$

Table 1
Variables for calculating forward penalties.

Variables	Definition
$a_i^{\min}$	Earliest allowed arrival time at a vertex i
$a_i^{\max}$	Arrival time at vertex i if as much fuel as possible has been recharged at preceding facilities
a_{ij}^{sl}	A possible slack between vertex i and vertex j that is happening due to time window limitations while traveling from vertex i to vertex j
a_i^{rt}	Inverse residual battery capacity at vertex i
a_{ij}^{add}	Additional fuel that has to be replenished at the preceding refueling facility to travel arc (i, j)
$\tilde{a}_i^{\min}$	Adjusted value of $a_i^{\min}$ to prevent a repeated penalization
$\tilde{a}_i^{\max}$	Adjusted value of $a_i^{\max}$ to prevent a repeated penalization

where

$$D = \begin{cases} \max\left\{0, a_v^{rt} + b_v^{rt} - B - \min\left\{B, \max\left\{0, b_v^{\min} - a_v^{\min}\right\}\right\}\right\} & \text{if } v \in F' \\ \max\{0, a_v^{rt} + b_v^{rt} - B - \min\{B, \min\{\max\{0, b_v^{\min} - a_v^{\min}\}, \\ \max\{0, a_v^{\max} - a_v^{\min}\} + \max\{0, b_v^{\min} - b_v^{\max}\}\}\}\} & \text{otherwise} \end{cases} \quad (44)$$

The concatenation of two partial routes, $\sigma_1 = \{0, \ldots, x\}$ and $\sigma_2 = \{y, \ldots, n+1\}$, that form $\sigma_c = \{0, \ldots, x, y, \ldots, n+1\}$ is obtained by first extending the variables of forward penalties to vertex y. The time window and battery penalty are able to be calculated by Eqs. (45) and (46) consecutively.

$$\begin{aligned} TW(\sigma_c) = {} & \overrightarrow{TW}(\sigma_1) + \overleftarrow{TW}(\sigma_2) + \max\left\{0, a_y^{\min}\right. \\ & \left. - l_y - \max\left\{0, a_y^{\min} - a_y^{\max}\right\}\right\} \\ & + \max\left\{0, \min\left\{l_v, \max\left\{e_y, a_y^{\min}\right\}\right\}\right\} \\ & - b_y^{\min} - \max\left\{b_y^{\max} - b_y^{\min}, 0\right\}\right\} \end{aligned} \quad (45)$$

$$BT(\sigma_c) = \overrightarrow{BT}(\sigma_1) + \overleftarrow{BT}(\sigma_2) + \max\left\{0, a_y^{\min} - a_y^{\max}\right\} + E \quad (46)$$

where:

$$E = \begin{cases} \max\left\{0, a_y^{rt} + b_y^{rt} - B - \min\left\{B, \max\left\{0, b_y^{\min} - a_y^{\min}\right\}\right\}\right\} & \text{if } y \in F' \\ \max\{0, a_y^{rt} + b_y^{rt} - B - \min\{B, \min\{\max\{0, b_y^{\min} - a_y^{\min}\}, \\ \max\{0, a_y^{\max} - a_y^{\min}\} + \max\{0, b_y^{\min} - b_y^{\max}\}\}\}\} & \text{otherwise} \end{cases} \quad (47)$$

If the electricity consumption model excludes cargo load from the consideration, then changes in battery capacity violation can be obtained in constant time for most neighborhood operators [46]. However, load is considered to be one factor that influences energy consumption in this work. This assumption will lead to a computational burden when concatenation operators in the algorithm are performed, because whenever a concatenation operator is performed, recalculations of time windows and battery forward and backward penalties should be conducted [30]. Hence, a surrogate violation concept, which was proposed by Goeke and Schneider [30], is adopted by assuming that the violations of vertices located prior to the point of change remain unmodified, and therefore the aforementioned equations to evaluate the concatenation will remain unchanged. Hereafter, the surrogate violation version of time windows will be mentioned as $\widetilde{TW}(\sigma)$ and the surrogate violation version of battery will be referred as $\widetilde{BT}(\sigma)$. The generalized cost function, which consists of surrogate violations, will be referred to as $\tilde{\lambda}_{gen}(\sigma)$. All the aforementioned variables in Table 1 should be updated after

applying the destroy phase, the repair phase, and the local search procedure. The update requires $O(n)$ time complexity where n is the number of nodes in σ.

3.4. Customer interval selection

The number of customers to remove is the beginning step to be performed before applying a removal heuristic. In the proposed ALNS, an adaptive mechanism to choose the number of customers to be removed is applied. Let $[\Omega_{\min}, \Omega_{\max}]$ be the allowable interval of number of customers to be removed. This interval is then divided into ω non-overlapping subintervals. ALNS then selects a subinterval according to the set of probability $\pi^c = \{\pi_1^c, \pi_2^c, \ldots, \pi_\omega^c\}$. Here, π^c is updated based on an adaptive mechanism that we will describe further in Section 3.10. From the selected subinterval, a random number δ of customers is determined.

3.5. Destroy operators

The developed ALNS employs five removal heuristics. The following subsections explain the heuristics. The input of the heuristics is δ, as described in Section 3.4. A removal heuristic aims to select δ customers to be removed from the solution. After one of these heuristics is applied to the solution, a list of removed customers is generated, and the solution becomes an incomplete solution.

3.5.1. Random removal

The random removal heuristic selects a number of customer nodes randomly and removes them from the solution. This heuristic runs the fastest among all utilized removal heuristics since it does not consider any criterion in removing nodes from a solution. The time complexity of random removal is $O(\delta)$.

3.5.2. Worst removal

This removal heuristic was introduced by Ropke and Pisinger [40] and aims to remove customer nodes that contribute significantly to the total cost if they are at their current position in a route. All customer nodes are sorted in descending order based on their cost value, which is determined by the change in $\tilde{\lambda}_{gen}(\sigma)$ that is calculated by removing the respective node. From the list, the node at position $\lfloor D.b^{\chi} \rfloor$ is selected, where D is total customers remaining in the solution, b is a uniform random number between 0 and 1, and χ is a parameter controlling the randomness of selecting a customer node from D that takes the value of 6 which is adopted from Goeke and Schneider [30]. Let c be the number of remaining customers in σ. In each iteration of removal process, we need $O(c)$ to calculate the objective change of removing each customer. In addition, we need $O(clogc)$ to perform the sorting procedure. After each iteration of the worst removal, we need to update the variables mentioned in Table 1, requiring time complexity of $O(n)$. Thus, the time complexity of this removal is $O((clogc + n + c)\delta)$.

3.5.3. Route removal

This removal was first proposed by Hemmelmayr et al. [48]. This heuristic works by randomly selecting a route and removing all nodes contained in the selected route. This heuristic is similar to random removal, but it performs at the route level. The computational time of the route removal heuristic depends on the number of nodes in the chosen route. Since this operator removes all customers in a selected route, the time complexity is $O(L)$ where L is the length of the route.

3.5.4. Shaw removal

This type of removal was introduced by Shaw [49] and later revised by Goeke and Schneider [30]. The basic concept is to remove customers that are similar to one another in regards to several criteria, and therefore they tend to be interchangeable when an insertion heuristic is implemented. These criteria are defined as relatedness. The relatedness measure consists of several components: geographical distance d_{ij}, difference in demand $|q_i - q_j|$, and difference of the earliest start of service $|e_i - e_j|$. The equation to calculate relatedness is as follows.

$$R(i,j) = \chi_d \frac{d_{ij}}{\max_{i,j\in V}(d_{ij})} + \chi_q \frac{|q_i - q_j|}{\max_{i\in V}(q_i) - \min_{i\in V}(q_i)} + \chi_e \frac{|e_i - e_j|}{\max_{i\in V}(e_i) - \min_{i\in V}(e_i)} \quad (48)$$

Here, χ_d, χ_q, and χ_e are parameters of each aforementioned criteria in the relatedness measure that take values of 6, 5, and 4, respectively. These values are all adopted from Goeke and Schneider [30].

One randomly removed customer is required in order to measure the relatedness of other customers. The customer at position $\lfloor D.b^{\chi} \rfloor$ is then chosen from the removal list, where D is total customers remaining in the solution, b is a uniform random number between 0 and 1, and χ is a parameter to control the randomness of a chosen position. After the first iteration, this heuristic will select a node from the removal list to measure the relatedness of other customers that have not been removed yet. This heuristic will terminate after δ customers have been removed from a solution. The time complexity of this operator is $O((logc + 1)\delta c)$.

3.5.5. Station vicinity

This removal heuristic was first proposed by Goeke and Schneider [30]. The idea of this heuristic is to re-order customer nodes that are located relatively close to recharging stations. To define the vicinity of a station, a radius value r is randomly chosen in the interval $\left[\chi_{radius}^{min}.\max_{i,j\in V}(d_{ij}), \chi_{radius}^{max}.\max_{i,j\in V}(d_{ij})\right]$, where $\chi_{radius}^{min} = 0.05$ and $\chi_{radius}^{min} = 0.15$. Next, a recharging station R is randomly chosen, and customer nodes located within a distance less than r are removed. The aforementioned step is repeated until δ customers are removed from a solution. The time complexity of this operator is $O(\delta)$.

3.6. Repair operators

After δ customers have been removed from a solution, the algorithm will proceed to the repair phase where an insertion heuristic is selected to repair the solution by inserting the removed δ customers into the incomplete solution obtained from the destroy phase. This phase aims to build a new better solution. The following sub-sections are devoted to describing the employed insertion heuristics during the repair phase.

3.6.1. Sequential insertion heuristics

This type of insertion heuristic works by iteratively performing the best possible insertion. For each customer node remaining on the removal list, the increase in the surrogate cost function for the insertion at every position of a route, $\tilde{\lambda}_{gen}(\sigma)$, is evaluated and the customer node is inserted to the position resulting in the minimum cost increase. We also utilize another version of sequential insertion heuristics by applying noise factor $x \in [\gamma_{min}, \gamma_{max}]$, where $\gamma_{min} = 0.8$ and $\gamma_{max} = 1.4$, to the calculated $\tilde{\lambda}_{gen}(\sigma)$ in the developed ALNS. The latter is referred to as perturbed sequential insertion heuristics and includes randomness into the original version, which leads to a higher probability to escape from local

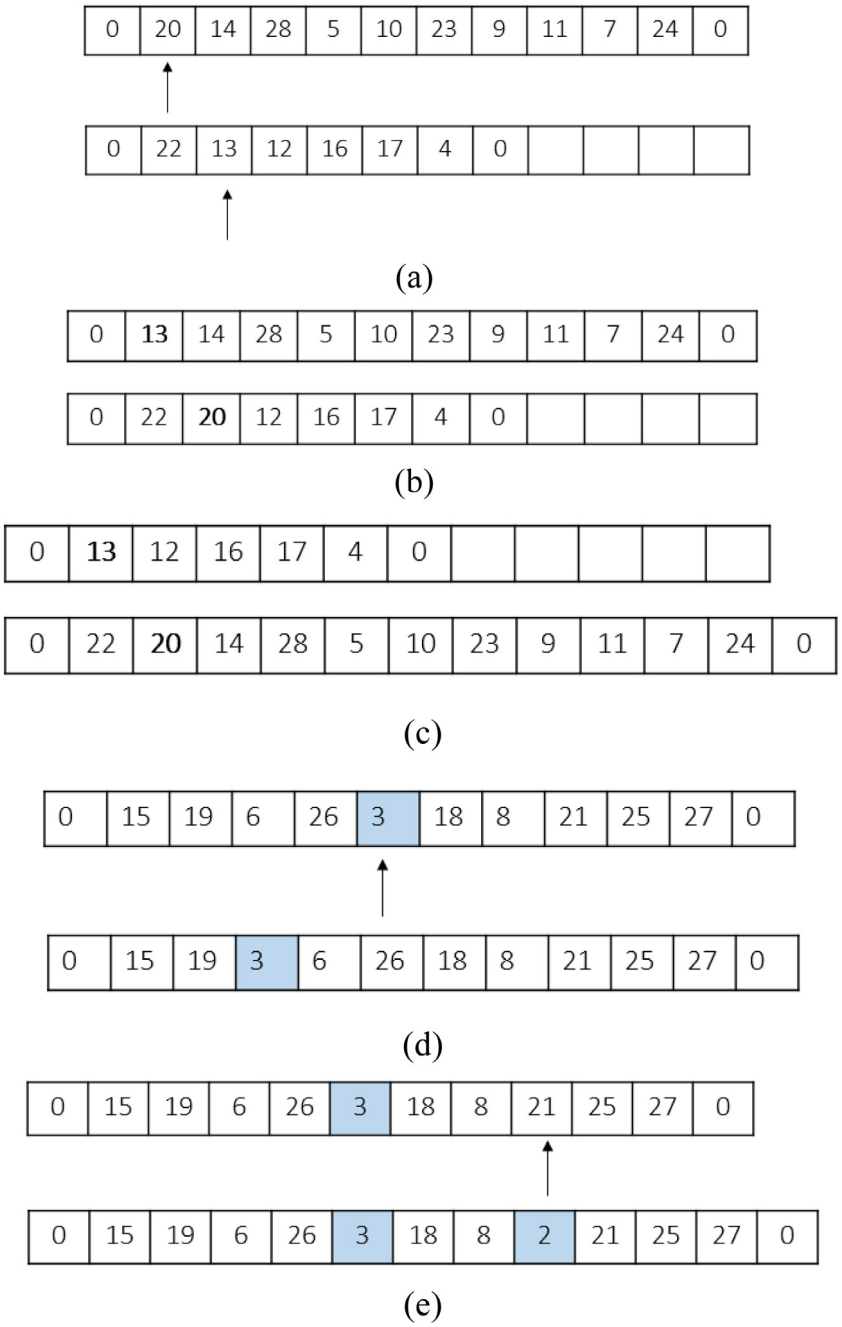


Fig. 2. (a) The reference solution before any neighborhood move is implemented, (b) the resulting solution after an exchange has been implemented to the reference solution, (c) the resulting solution after 2-opt* has been applied to the reference solution, (d) illustration of implementing the relocate move, (e) an illustration of implementing station-in move.

optima. The time complexity of performing this insertion heuristic depends on the number of customers in the removal list δ, the number of insertion positions L_i, and the time complexity of evaluating the insertion itself. As mentioned in Section 3.3, the time complexity of the concatenation operators employed in the insertion procedure to evaluate the cost of insertion is $O(1)$. Consequently, the time complexity of sequential insertion heuristics is $O(\delta L_i)$ in each iteration of insertion.

3.6.2. Regret insertion heuristics

This type of insertion was described by Ropke and Pisinger [40] which later modified by Liu et al. [43]. A k-regret value for each customer node is calculated as the difference between the insertion cost in the best position and that cost in the k-best position. The cost of insertion is evaluated by $\tilde{\lambda}_{gen}(\sigma)$. This research applies the regret-2 and regret-3 insertion heuristics. Similar to sequential insertion heuristics, the perturbed versions of these regret insertion heuristics are embedded as options in the repair phase. Let L_i be the number of insertion positions for a particular node. For each evaluated node in the removal list, the required time complexity is $O(L_i)$ since there are L_i evaluations of objective change. Then, we perform sorting procedure on the obtained objective changes to find the k best positions, requiring $O(L_i log L_i)$. Thus, the time complexity of regret insertion heuristics is $O((L_i + (L_i log L_i))\delta)$ in each iteration of insertion.

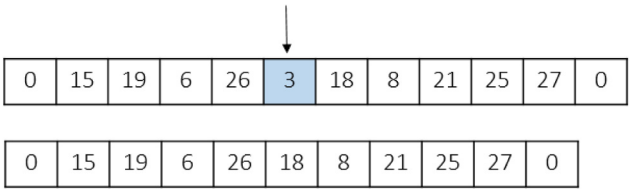


Fig. 3. An illustration of removing the recharging station from a solution.

3.7. Local search procedure

In order to enhance the solution quality, a local search (LS) is applied. A composite neighborhood comprised of several moves, i.e. exchange, 2-opt*, relocate, and station-in, are utilized in LS. Fig. 2.b to e illustrate how the neighborhood moves respectively work, whereas Fig. 2.a depicts the original solution before exchange and 2-opt* are implemented.

Exchange and 2-opt* are applied for inter-route moves, while relocate is applied for intra- and inter-route moves, and they only deal with recharging station nodes. Moreover, 2-opt* is not allowed to be applied between ICV and EV routes. Station-in is a special local search operator first proposed by Schneider et al. [19] and aims to insert a recharging station into EV routes.

The neighborhood size of the exchange move is $O(n_e^2)$, while the neighborhood size of the 2-opt* move for ICV routes and EV routes are $O(n_{ICV}^2)$ and $O(n_{EV}^2)$, respectively [50]. n_{ICV} and n_{EV} represent the number of nodes visited in ICV routes and EV routes, respectively, while n_e denotes the number of nodes to be exchanged in the solution. Since the time complexity of evaluating the feasibility of inter-route moves is $O(1)$ as explained in Section 3.3, the time complexity of performing exchange and 2-opt* are consequently the same as their neighborhood sizes. The time complexity of relocate is $O(n_{EV})$ since it evaluates all available positions in EV routes and the worst-time complexity of station-in is $O(kn_{EV})$, where k denotes the number of available recharging stations.

When the moves generated by LS result in an infeasible solution, they will be automatically discarded. A surrogate cost function, $\tilde{\lambda}_{gen}(.)$, is utilized to evaluate the moves. These strategies are taken in order to reduce the computational burden of performing a local search. In each iteration of LS, a list $\mathbb{M}$ containing the best ε solutions generated by the composite neighborhood is generated. An exact evaluation of the objective value, $\lambda_{gen}(.)$, is then performed on each solution in $\mathbb{M}$. The time complexity of exact evaluation of a solution is $O(n)$. Thus, this step requires a $O(\varepsilon n)$ time to complete. Next, the solution with the best λ_{gen} on the list is applied to the solution. LS stops when no further improvement is found by the neighborhood.

3.8. Dynamic programming to solve the optimal placement of a recharging station

In order to improve the solution quality of routes traveled by EVs, dynamic programming (DP) is utilized after a number of search steps to identify optimal placement of recharging station visits for the routes traveled by EVs. The DP utilized in this work is adopted from Schiffer and Walther [46] with a modification that will be explained in the following paragraph. DP starts by removing all the recharging station visits of a particular route. Fig. 3 depicts a solution that initially consists of a visit to recharging station 3 and later is dropped, leading to a solution solely consisting of customer node visits.

After removing recharging stations in the solution, the dynamic programming procedure is executed. The procedure tries to create a path by inserting each possible recharging station after each customer node in the solution, starting from the first customer node. A path is further extended as long as it remains feasible and is not dominated by another path. Resource extension functions (REFs) are used for feasibility and dominance checks of each developed path. By doing so, optimal recharging station visits for the considered route are identified. Based on the aforementioned explanation, the worst-time complexity of this procedure is $O(k^{L_{route}})$, where k and L_{route} denote the number of available recharging stations and the number of insertion positions in the evaluated EV route, respectively.

The original idea of REFs was proposed by Desaulniers et al. [51] and extended by Schiffer and Walther [46]. A modification is made to adapt DP to our problem by defining $h_{ij} = r.b_{ij}^{k_E}(u_{ij}^{k_E})$. The REFs necessary to extend a path from node i to node j are written in Eqs. (49)–(55). Table 2 describes the label components.

Table 2
A description of label component for REFs.

T_i^{cost}	Cost of path
T_i^{F}	Number of recharging station visits on path
T_i^{tMin}	Earliest arrival time on node i without running out of electric energy
T_i^{tMax}	Latest arrival time on node i recharging as much as possible at preceding recharging stations, without violating time windows
T_i^{rtMax}	Maximal amount of energy that can be recharged (expressed in time unit)

$$T_j^{\text{cost}} = T_i^{\text{cost}} + d_{ij} \tag{49}$$

$$T_j^{\text{F}} = T_i^{\text{F}} + \begin{cases} 1 \text{ if } j \in F', \\ 0 \text{ else,} \end{cases} \tag{50}$$

$$T_j^{\text{tMin}} = \begin{cases} \max\left\{e_j, T_i^{\text{tMin}} + t_{ij}\right\} \text{ if } T_i^{\text{F}} = 0, \\ \max\left\{e_j, T_i^{\text{tMin}} + t_{ij}\right\} + X_{ij}\left(T_i^{\text{tMin}}, T_i^{\text{tMax}}\right) \text{ else,} \end{cases} \tag{51}$$

$$T_j^{\text{tMax}} = \begin{cases} \min\left\{l_j, \max\left\{e_j, T_i^{\text{tMin}} + \max\left\{0, T_i^{\text{rtMax}}\right\} + t_{ij}\right\}\right\} \text{ if } i \in F' \\ \min\left\{l_j, \max\left\{e_j, T_i^{\text{tMax}} + t_{ij}\right\}\right\} \text{ else,} \end{cases} \tag{52}$$

$$T_j^{\text{rtMax}} = \begin{cases} T_i^{\text{rtMax}} + h_{ij} \text{ if } T_i^{\text{F}} = 0, \\ \min\left\{H, \max\left\{0, T_i^{\text{rtMax}} - S_{ij}\left(T_i^{\text{tMin}}\right)\right\} + h_{ij}\right\} \text{ if } i \in F', \\ \min\left\{H, \max\left\{0, T_i^{\text{tMax}} - \min\left\{S_{ij}\left(T_i^{\text{tMin}}\right), T_i^{\text{tMax}} - T_i^{\text{tMin}}\right\}\right\} + h_{ij}\right\} \\ \text{else,} \end{cases} \tag{53}$$

$$X_{ij}\left(T_i^{\text{tMin}}, T_i^{\text{tMax}}\right) = \begin{cases} \max\left\{0, \max\left\{0, T_i^{\text{rtMax}} - S_{ij}\left(T_i^{\text{tMin}}\right)\right\} + h_{ij} - H\right\} \text{ if } i \in F' \\ \max\left\{0, \max\left\{0, T_i^{\text{rtMax}} - \min\left\{S_{ij}\left(T_i^{\text{tMin}}\right), T_i^{\text{tMax}} - T_i^{\text{tMin}}\right\}\right\} + h_{ij} - H\right\} \text{ else,} \end{cases} \tag{54}$$

$$S_{ij}\left(T_i^{\text{tMin}}\right) = \max\left\{0, e_j - \left(T_i^{\text{tMin}} + t_{ij}\right)\right\} \tag{55}$$

A label $L_j = \left(T_j^{\text{cost}}, T_j^{\text{F}}, T_j^{\text{tMin}}, T_j^{\text{tMax}}, T_j^{\text{rtMax}}\right)$ is feasible if the following requirement holds:

$$\left(T_j^{\text{tMin}} \le l_j\right) \wedge \left(T_j^{\text{tMin}} \le T_j^{\text{tMax}}\right) \wedge \left(T_j^{\text{rtMax}} \le H\right) \tag{56}$$

For the dominance check, a label L_2 is dominated by L_1 and thus neglected if:

$$T_1^{\text{cost}} \le T_2^{\text{cost}}, \tag{57}$$

$$T_1^{\text{rtMax}} - \left(T_1^{\text{tMax}} - T_1^{\text{tMin}}\right) \le T_2^{\text{rtMax}} - \left(T_2^{\text{tMax}} - T_2^{\text{tMin}}\right), \tag{58}$$

$$T_1^{\text{rtMax}} - \left(T_2^{\text{tMin}} - T_1^{\text{tMin}}\right) \le T_2^{\text{rtMax}}. \tag{59}$$

3.9. Generating a feasible solution procedure

As an effort to obtain feasible solutions, an approach proposed by Vidal et al. [52] is employed in this algorithm; i.e. we use the local search procedure with penalty weights multiplied by

100. If no feasible solution is found after applying the procedure, then penalty weights are multiplied by 10 and a local search is performed again. No further steps are conducted to obtain a feasible solution if the second step fails to find a feasible solution.

3.10. Adaptive mechanism

During the destroy and repair phase, heuristics are selected based on their probability values. This section describes the adaptive mechanism of the destroy and repair phase in detail, and a similar procedure is also applied in order to select the interval of number of customers to be removed, as described in Section 3.4.

The probability values assigned to each heuristic reflect how successful each heuristic improves the solution during iterations of ALNS. Here, D and R are a set of removal and insertion heuristics, as briefly explained in Section 3. Let $D = \{D_1, D_2, \ldots, D_{ND}\}$ and $R = \{R_1, R_2, \ldots, R_{NR}\}$, where ND and NR represent the number of available removal heuristics and available insertion heuristics, consecutively. Additionally, $P(D_i)$ and $P(R_i)$ represent the probability of choosing D_i and R_i, respectively, where D_i is the removal heuristic i and R_i is the insertion heuristic i. Both D_i and R_i depend on the weight of each heuristic, $\pi = \{\pi^D \cup \pi^R\}$, where π^D and π^R consecutively represent a set of weights of removal heuristics and insertion heuristics where $\pi^D = \{\pi_1^D, \pi_2^D, \ldots, \pi_{ND}^D\}$ and $\pi^R = \{\pi_1^R, \pi_2^R, \ldots, \pi_{NR}^R\}$. These weights are calculated using Eqs. (60) and (61).

$$P(D_i) = \frac{\pi_i^D}{\sum_{j=1}^{ND} \pi_j^D} \tag{60}$$

$$P(R_i) = \frac{\pi_i^R}{\sum_{j=1}^{NR} \pi_j^R} \tag{61}$$

These weights are always updated every η^s iterations. In order to calculate a weight of a heuristic, a score is collected after a heuristic performs, which is denoted as σ_i^D and σ_i^R where σ_i^D is the score of removal heuristic i and σ_i^R is the score of insertion heuristic i. The score is updated based on a scoring system $\left(\varepsilon^f, \varepsilon^b, \varepsilon^i, \varepsilon^w\right)$. After performing a preliminary study, we set $\varepsilon^f = 15$, $\varepsilon^b = 9$, $\varepsilon^i = 4$, and $\varepsilon^w = 1$. If a new best feasible solution is found, then ε^f is added to the total score of the utilized heuristic. If a new best solution is found, then ε^b is added. If the new solution improves the current one, then ε^i is added, and if the new solution results in a worse solution yet it is still accepted by the SA acceptance mechanism, then ε^w is added. After η^s iterations have passed, the weight of each heuristic is updated using Eqs. (62) and (63). After the weight is updated, all statistics in Eqs. (62) and (63) are set back to zero.

$$\pi_i^D = \varphi \frac{\sigma_i^D}{\chi_i^D} + (1-\varphi)\pi_i^D \tag{62}$$

$$\pi_i^R = \varphi \frac{\sigma_i^R}{\chi_i^R} + (1-\varphi)\pi_i^R \tag{63}$$

where:
φ is the smoothing factor and ranges between 0 and 1;
χ_i^D is the frequency of utilizing removal heuristic i;
χ_i^R is the frequency of utilizing insertion heuristic i.

3.11. Simulated annealing acceptance mechanism

The concept of the Simulated Annealing (SA) mechanism is to not always reject worse solutions. We ccan accept a worse solution with a particular probability depending on (i) the difference between cost values of the new and the current solutions and (ii) number of iterations that has been made.

A careful treatment needs to be conducted since we allow for a dynamic adjustment of penalty factors. The main impact is that the cost value of a solution strongly depends on the current values of the penalty factors. Consequently, evaluating two solutions by utilizing two different sets of penalty values can be misleading, because the cost difference may be merely caused by the difference in penalty values.

Let $\left(\gamma_{cap}^{\sigma}, \gamma_{tw}^{\sigma}, \gamma_{batt}^{\sigma}\right)$ be the vector of penalty values employed for evaluating $\lambda_{gen}(\sigma)$ and $\left(\gamma_{cap}^{\sigma'}, \gamma_{tw}^{\sigma'}, \gamma_{batt}^{\sigma'}\right)$ be the penalty values used for evaluating $\lambda_{gen}(\sigma')$ at the time of solution creation. In order to make two solution, σ and σ', comparable, revised penalty values are calculated using Eq. (64) prior to calculating the acceptance probability.

$$\left(\gamma_{cap}^{eval}, \gamma_{tw}^{eval}, \gamma_{batt}^{eval}\right) = \left(\frac{1}{2}\cdot\left(\gamma_{cap}^{S} + \gamma_{cap}^{S'}\right) + \frac{1}{2}\cdot\left(\gamma_{tw}^{S} + \gamma_{tw}^{S'}\right) + \frac{1}{2}\cdot\left(\gamma_{batt}^{S} + \gamma_{batt}^{S'}\right)\right) \tag{64}$$

The cost difference between two solutions is then calculated based on the revised penalty values using Eq. (65), and deteriorating solutions are accepted with an acceptance probability calculated by using Eq. (66).

$$\Delta(\sigma', \sigma) = \frac{\lambda_{gen}^{eval}(\sigma') - \lambda_{gen}^{eval}(\sigma)}{\lambda_{gen}^{eval}(\sigma)} \tag{65}$$

$$p(\sigma, \sigma', T) = e^{\frac{-\Delta(\sigma', \sigma)}{T}} \tag{66}$$

A value for the parameters related to the SA mechanism is set by following Goeke and Schneider (2015). The initial temperature is determined so that a solution that is worse than the initial solution by 50% is accepted with a probability of 50%. The temperature is then decreased by a constant factor such that the temperature is below $T_{threshold} = 0.0001$ in the last 20% of iterations.

4. Computational results

Section 4 focuses on describing the performance of the proposed ALNS algorithm and the benefit of dealing with GMFVRP-REC-PR in terms of carbon emission reduction. The proposed ALNS algorithm was coded in C++ and tested on a computer with an Intel Core i7-7700 CPU @3.60 GHz processor and 8.00 GB RAM.

4.1. Test instances

The benchmark instances are obtained from Goeke and Schneider [30]. These instances were originally proposed by Demir et al. [53] for the pollution routing problem (PRP). Each instance consists of a set of customers whose locations represent real cities in the UK, and each customer has an amount of demand and particular time windows that are all random. In total, there are nine benchmark instance sets, varying in the number of customers (10 to 200 customers). Each instance set contains 20 different instances. A modification was made by Goeke and Schneider [30] in a way that there is only one value of vehicle speed, 90 kilometers per hour, instead of varying speed values in PRP. In addition, Goeke and Schneider [30] generated a number of recharging stations, i.e. $\lfloor 0.1.|N| \rfloor$, where N is number of customers, and the location of each recharging station is generated randomly. The variables in the realistic energy consumption are obtained from Demir et al. [53]. The value of FE, fuel conversion factor, is 2.6 and obtained from Macrina et al. [33] by rounding up to one number after the decimal point. The parameters of the realistic energy consumption model are obtained from Goeke and Schneider [30].

Table 3
The considered parameters in the parameter selection phase. The bold values are the best combination among all available combinations under a 2^{k-2} factorial design experiment that results in the lowest average cost.

Parameter	Values
η_{noi}^{max}	(800, **1400**)
η^{res}	(**400**, 600)
η^{S}	(**50**, 80)
η^{p}	(10, **30**)
φ	(0.6, **0.8**)
γ_{cap}^{init}, γ_{tw}^{init}, γ_{batt}^{init}	(200, **400**)
Ω	(**1.15**, 1.7)

4.2. Parameter calibration

Before applying our developed ALNS algorithm to the test instances, we conduct the parameters' selection procedure. The aim of the parameter selection is to find the combination of considered parameters among the available options that result in the best performance of the algorithm. We utilize a 2^{k-2} factorial design to select the best combination of parameters and randomly choose 18 out of 180 instances to be solved, or two problems from each of nine test instance sets. ALNS is then run five times for each instance.

The parameters considered in this phase are: (1) Number of maximum non-improvement iterations (η_{noi}^{max}), (2) Number of necessary iterations to perform a restart strategy (η^{res}), (3) Number of necessary iterations for updating the probability of selecting removal and insertion heuristics (η^{S}), (4) Number of necessary iterations for updating the penalty values (η^{p}), (5) The smoothing factor (φ), (6) Initial penalty values $\left(\gamma_{cap}^{init}, \gamma_{tw}^{init}, \gamma_{batt}^{init}\right)$, and (7) Multiplication factor of penalty values (Ω). Table 3 contains the considered parameters of ALNS.

4.3. Performance of ALNS on solving E-VRPTWMF instances

We then apply our proposed ALNS to solve the benchmark test instances proposed by Goeke and Schneider [30]. The problem addressed by Goeke and Schneider [30] is E-VRPTWMF and

Table 4
Comparison summary of solutions obtained by Goeke and Schneider [30] and our developed ALNS.

Instance set	Goeke and Schneider (2015)				ALNS				Average gap (%)
	ICV	EV	Average cost	Average CPU time (mins)	ICV	EV	Average cost	Average CPU time (mins)	
Instance set 1: 10 customers	20	20	479.612	0.034	20	20	479.659	0.008	0.012
Instance set 2: 15 customers	33	18	639.51	0.078	34	17	626.46	0.022	−0.901
Instance set 3: 20 customers	40	21	787.579	0.156	40	21	788.849	0.047	0.148
Instance set 4: 25 customers	40	33	824.448	0.278	40	33	827.367	0.126	0.328
Instance set 5: 50 customers	78	63	1414.383	1.455	78	63	1427.33	1.368	0.888
Instance set 6: 75 customers	108	100	2041.475	2.876	109	99	2044.487	3.954	0.896
Instance set 7: 100 customers	139	129	2564.567	6.705	139	128	2594.574	11.099	1.192
Instance set 8: 150 customers	204	197	3639.389	11.912	204	197	3652.167	30.784	0.405
Instance set 9: 200 customers	**271**	**256**	**4517.461**	**16.715**	**271**	**254**	**4473.824**	**91.94**	−0.988

Table 5
Computational results for small GMFVRP-REC-PR instances (10 and 15 customers).

Instance	ICV	EV	CPLEX		ALNS		Instance	ICV	EV	CPLEX		ALNS	
			Total cost	CPU Time(mins)	%Gap (cost)	%Gap (time)				Total cost	CPU Time(mins)	%Gap (cost)	%Gap (time)
E-UK10_01	1	1	544.319	25.031	0.00	−96.00	E-UK15_01	2	0	1071.18	35.474	0.00	−97.71
E-UK10_02	1	1	692.87	10	0.00	−90.41	E-UK15_02	1	1	654.05	306.2	0.00	−99.39
E-UK10_03	1	1	664.81	18.859	0.00	−95.55	E-UK15_03	2	1	997.13	3294.88	0.00	−99.95
E-UK10_04	1	1	689.8	22.547	0.00	−96.11	E-UK15_04[a]	2	1	1014.06	7361.45	0.74	−99.95
E-UK10_05	1	1	575.95	11.937	0.00	−93.06	E-UK15_05	2	0	1094.87	3.026	0.00	−63.07
E-UK10_06	1	1	805.95	69.576	0.00	−98.74	E-UK15_06[a]	**2**	**1**	**806.71**	**7629.62**	−3.23	−99.98
E-UK10_07	1	1	729.888	226.295	2.65	−99.59	E-UK15_07	2	1	865.24	2649.22	0.00	−99.94
E-UK10_08	1	1	779.61	23.946	0.00	−95.85	E-UK15_08[a]	1	1	538.34	7208.15	0.00	−99.95
E-UK10_09	1	1	644.87	131.134	0.61	−99.74	E-UK15_09	2	1	866.07	2019.9	0.00	−99.85
E-UK10_10	1	1	755.47	297.275	0.11	−99.85	E-UK15_10	1	1	769.57	3210.42	0.00	−99.94
E-UK10_11	1	1	987.842	10.483	0.00	−91.93	E-UK15_11	2	0	972.35	4337	0.00	−99.98
E-UK10_12	1	1	578.846	47.377	0.00	−98.41	E-UK15_12	2	1	1061.66	1174.41	0.00	−99.86
E-UK10_13	1	1	756.9	85.66	0.00	−99.32	E-UK15_13[a]	**2**	**1**	**882.072**	**7300.38**	−0.29	−99.99
E-UK10_14	1	1	578.12	38.797	0.00	−99.18	E-UK15_14[a]	**2**	**1**	**1221.93**	**7346.17**	−0.17	−99.98
E-UK10_15	1	1	343.64	45.661	0.00	−99.09	E-UK15_15[a]	1	1	731.7	7228.24	0.00	−99.99
E-UK10_16	1	1	548.095	28.923	0.00	−96.55	E-UK15_16[a]	1	1	734.15	7209.13	0.00	−99.98
E-UK10_17	1	1	569.836	26.083	0.00	−98.41	E-UK15_17[a]	2	1	996.58	7249.99	0.00	−99.99
E-UK10_18	1	1	535.95	200.321	0.00	−99.76	E-UK15_18	2	1	1101.34	4315.18	0.00	−99.97
E-UK10_19	1	1	575.138	22.604	0.00	−98.17	E-UK15_19	1	1	498.81	119.59	0.00	−97.25
E-UK10_20	1	1	495.153	65.505	0.00	−98.53	E-UK15_20	2	1	687.24	4046.73	0.00	−99.97
Average					**0.17**	**−97.21**	**Average**					**−0.15**	**−97.83**

[a]Indicates that the instance is not solved to optimality.
Bold values indicate that the ALNS can obtain better solution compared to CPLEX.

Table 6
Computational results of the propose ALNS on solving the GMFVRP-REC-PR instances.

Instance	m_{IC}	m_E	Best Cost	Stdev	CV	CPU Time	Instance	m_{IC}	m_E	Best Cost	Stdev	CV	CPU Time	Instance	m_{IC}	m_E	Best Cost	Stdev	CV	CPU Time
E-UK10_01	1	1	544.32	0.00	0.00	0.02	E-UK25_01	2	1	915.51	2.00	0.22	0.14	E-UK100_01	7	7	3565.07	36.89	1.02	21.02
E-UK10_02	1	1	692.87	0.29	0.04	0.02	E-UK25_02	2	2	1031.62	3.25	0.31	0.21	E-UK100_02	7	6	3551.73	49.04	1.36	10.75
E-UK10_03	1	1	664.81	0.00	0.00	0.01	E-UK25_03	2	1	598.60	9.89	1.63	0.17	E-UK100_03	7	6	3230.70	48.66	1.49	25.33
E-UK10_04	1	1	689.80	0.36	0.05	0.01	E-UK25_04	2	1	856.11	0.00	0.00	0.10	E-UK100_04	7	7	3052.26	61.52	1.97	21.97
E-UK10_05	1	1	575.95	2.18	0.38	0.01	E-UK25_05	2	2	997.76	16.81	1.65	0.15	E-UK100_05	7	7	2730.19	30.42	1.10	13.47
E-UK10_06	1	1	805.95	0.65	0.08	0.01	E-UK25_06	2	2	919.53	22.56	2.41	0.12	E-UK100_06	7	7	3563.61	18.17	0.51	8.58
E-UK10_07	1	1	749.22	0.69	0.09	0.02	E-UK25_07	2	1	1083.86	6.62	0.61	0.17	E-UK100_07	6	6	3058.37	66.32	2.12	21.27
E-UK10_08	1	1	779.61	0.00	0.00	0.02	E-UK25_08	2	1	1217.34	21.23	1.71	0.07	E-UK100_08	7	6	3375.74	51.37	1.50	14.28
E-UK10_09	1	1	648.78	0.00	0.00	0.01	E-UK25_09	2	2	890.78	0.08	0.01	0.11	E-UK100_09	7	6	2759.51	12.78	0.46	27.33
E-UK10_10	1	1	756.31	2.22	0.29	0.01	E-UK25_10	2	2	1121.35	1.72	0.15	0.23	E-UK100_10	6	6	3119.50	16.84	0.54	18.91
E-UK10_11	1	1	987.85	0.00	0.00	0.01	E-UK25_11	2	2	1140.37	13.05	1.13	0.24	E-UK100_11	7	7	3596.45	60.91	1.65	13.71
E-UK10_12	1	1	578.85	0.00	0.00	0.01	E-UK25_12	2	2	1354.48	2.44	0.18	0.09	E-UK100_12	6	6	3008.03	67.68	2.18	11.32
E-UK10_13	1	1	756.91	0.00	0.00	0.01	E-UK25_13	2	2	605.13	1.70	0.28	0.11	E-UK100_13	7	6	3466.39	55.44	1.56	17.57
E-UK10_14	1	1	578.13	0.00	0.00	0.01	E-UK25_14	2	2	1240.10	34.79	2.76	0.11	E-UK100_14	7	7	3651.74	36.23	0.98	24.26
E-UK10_15	1	1	343.64	0.20	0.06	0.01	E-UK25_15	2	1	1249.93	3.32	0.27	0.04	E-UK100_15	8	7	3856.20	83.29	2.11	18.46
E-UK10_16	1	1	548.10	0.00	0.00	0.02	E-UK25_16	2	2	1066.19	10.67	0.99	0.12	E-UK100_16	6	6	2840.79	18.69	0.65	18.33
E-UK10_17	1	1	569.84	0.00	0.00	0.01	E-UK25_17	2	2	1659.60	6.58	0.41	0.04	E-UK100_17	8	7	3822.75	91.08	2.32	12.34
E-UK10_18	1	1	535.95	0.00	0.00	0.01	E-UK25_18	2	1	1292.06	5.72	0.44	0.12	E-UK100_18	7	6	3137.17	15.97	0.50	16.63
E-UK10_19	1	1	575.14	0.00	0.00	0.01	E-UK25_19	2	2	1429.98	24.23	1.67	0.13	E-UK100_19	7	6	2933.26	32.35	1.10	26.80
E-UK10_20	1	1	495.16	0.00	0.00	0.02	E-UK25_20	2	2	1054.68	9.13	0.86	0.17	E-UK100_20	7	7	3783.91	72.86	1.89	9.07
E-UK15_01	2	0	1071.18	0.00	0.00	0.01	E-UK50_01	4	3	1767.69	32.65	1.81	2.23	E-UK150_01	10	10	4097.71	68.71	1.65	29.61
E-UK15_02	1	1	654.05	0.33	0.05	0.03	E-UK50_02	4	3	1887.33	27.36	1.42	1.35	E-UK150_02	10	10	4948.88	131.00	2.60	72.14
E-UK15_03	2	1	997.13	0.00	0.00	0.03	E-UK50_03	4	3	1881.52	19.65	1.03	2.98	E-UK150_03	10	9	4047.67	94.37	2.28	73.56
E-UK15_04	2	1	1014.06	0.00	0.00	0.06	E-UK50_04	4	4	2208.11	34.82	1.55	1.99	E-UK150_04	11	10	4673.19	25.32	0.54	94.39
E-UK15_05	2	0	1094.88	0.00	0.00	0.02	E-UK50_05	3	3	2042.18	24.33	1.17	1.03	E-UK150_05	10	10	4283.76	102.46	2.34	55.38
E-UK15_06	2	1	806.71	14.26	1.78	0.02	E-UK50_06	4	4	1596.36	25.95	1.59	2.83	E-UK150_06	11	10	4288.05	55.55	1.28	52.14
E-UK15_07	2	1	865.24	0.00	0.00	0.03	E-UK50_07	4	3	1466.89	17.64	1.18	2.56	E-UK150_07	11	10	5005.33	81.34	1.59	42.82
E-UK15_08	1	1	538.34	0.44	0.08	0.06	E-UK50_08	4	3	1715.72	21.72	1.25	0.70	E-UK150_08	10	10	4327.06	69.17	1.57	109.27
E-UK15_09	2	1	866.07	0.46	0.05	0.05	E-UK50_09	4	3	2117.84	36.00	1.65	1.59	E-UK150_09	10	10	4673.95	43.57	0.92	47.52
E-UK15_10	1	1	769.57	0.00	0.00	0.03	E-UK50_10	4	3	2048.91	22.19	1.07	2.61	E-UK150_10	10	10	4562.50	31.80	0.69	57.36
E-UK15_11	2	0	972.35	0.00	0.00	0.01	E-UK50_11	4	3	1992.11	32.54	1.61	0.65	E-UK150_11	10	10	5070.56	132.82	2.58	40.88
E-UK15_12	2	1	1061.66	0.00	0.00	0.03	E-UK50_12	4	3	1731.31	5.93	0.34	0.78	E-UK150_12	11	10	4982.46	68.91	1.36	43.70
E-UK15_13	2	1	882.07	1.52	0.17	0.02	E-UK50_13	4	3	1744.83	32.02	1.80	1.97	E-UK150_13	10	9	4767.65	69.65	1.44	77.08
E-UK15_14	2	1	1221.93	7.80	0.64	0.02	E-UK50_14	4	3	2064.35	8.05	0.39	1.13	E-UK150_14	10	10	4801.21	62.29	1.27	48.44
E-UK15_15	1	1	731.70	1.70	0.23	0.02	E-UK50_15	3	3	1783.37	23.30	1.30	2.10	E-UK150_15	10	9	4123.96	76.46	1.81	55.93
E-UK15_16	1	1	734.15	0.00	0.00	0.02	E-UK50_16	4	3	1684.68	24.01	1.40	1.37	E-UK150_16	10	10	4936.33	95.25	1.90	46.00
E-UK15_17	2	1	996.58	0.00	0.00	0.02	E-UK50_17	4	3	1161.70	15.25	1.30	1.92	E-UK150_17	10	10	4738.68	36.42	0.76	91.71
E-UK15_18	2	1	1101.34	14.74	1.31	0.02	E-UK50_18	4	4	1968.20	39.52	1.96	1.59	E-UK150_18	10	10	4851.65	74.65	1.51	52.23
E-UK15_19	1	1	498.81	0.00	0.00	0.05	E-UK50_19	4	3	1788.58	16.40	0.91	1.22	E-UK150_19	10	10	5437.31	83.80	1.52	22.21
E-UK15_20	2	1	687.24	1.55	0.23	0.02	E-UK50_20	4	3	2088.26	32.13	1.51	1.03	E-UK150_20	10	10	5071.47	18.08	0.36	63.58
E-UK20_01	2	1	1071.08	0.00	0.00	0.05	E-UK75_01	6	5	2913.18	30.85	1.04	4.14	E-UK200_01	14	14	5798.64	82.10	1.39	173.71
E-UK20_02	2	1	1154.29	23.69	2.02	0.04	E-UK75_02	6	5	2387.97	19.59	0.81	5.07	E-UK200_02	12	12	5528.58	59.75	1.07	72.38
E-UK20_03	2	1	599.16	0.00	0.00	0.06	E-UK75_03	5	5	2510.62	20.86	0.83	7.08	E-UK200_03	14	13	5681.30	18.09	0.32	37.76
E-UK20_04	2	1	1080.80	2.06	0.19	0.09	E-UK75_04	6	5	2261.07	22.46	0.98	5.08	E-UK200_04	13	13	5084.53	40.96	0.79	94.81
E-UK20_05	2	1	1011.65	12.30	1.21	0.04	E-UK75_05	5	5	2673.89	36.53	1.35	6.12	E-UK200_05	14	13	6036.57	91.61	1.48	108.43
E-UK20_06	2	1	1137.83	1.66	0.15	0.06	E-UK75_06	6	5	2798.60	31.65	1.12	3.45	E-UK200_06	13	13	4904.51	87.39	1.74	139.46
E-UK20_07	2	1	705.06	2.54	0.36	0.07	E-UK75_07	6	5	2810.67	7.79	0.28	3.09	E-UK200_07	14	13	5497.56	112.19	1.99	158.71
E-UK20_08	2	1	888.43	0.00	0.00	0.06	E-UK75_08	6	5	2816.02	38.12	1.33	6.37	E-UK200_08	14	13	5842.03	138.44	2.32	135.57
E-UK20_09	2	1	1148.65	13.14	1.14	0.03	E-UK75_09	5	5	2760.46	25.43	0.91	6.93	E-UK200_09	13	12	5086.59	59.20	1.15	217.20
E-UK20_10	2	1	914.49	0.32	0.04	0.08	E-UK75_10	**5**	**5**	2881.38	17.65	0.61	10.78	E-UK200_10	13	14	6200.38	29.75	0.48	104.63
E-UK20_11	2	1	1230.51	3.33	0.27	0.06	E-UK75_11	5	5	1860.18	34.99	1.84	13.12	E-UK200_11	14	13	5183.09	97.45	1.85	141.28
E-UK20_12	2	1	1089.74	1.38	0.13	0.05	E-UK75_12	5	5	2489.64	21.28	0.84	8.14	E-UK200_12	13	12	6094.57	52.52	0.85	150.32
E-UK20_13	2	1	1070.17	11.90	1.11	0.04	E-UK75_13	5	5	2951.43	34.91	1.17	2.72	E-UK200_13	13	12	6076.72	68.49	1.11	108.79
E-UK20_14	2	2	1345.53	0.00	0.00	0.08	E-UK75_14	5	5	2756.39	35.63	1.27	7.04	E-UK200_14	14	13	5547.18	117.10	2.06	126.56
E-UK20_15	2	1	1075.22	9.52	0.87	0.06	E-UK75_15	5	5	2939.96	44.53	1.50	3.08	E-UK200_15	13	12	6040.53	116.06	1.89	69.27
E-UK20_16	2	1	1113.41	0.74	0.07	0.05	E-UK75_16	5	5	2723.48	21.60	0.79	1.54	E-UK200_16	14	13	5624.33	107.22	1.87	152.27
E-UK20_17	2	1	1196.03	6.05	0.50	0.11	E-UK75_17	6	5	2604.64	29.01	1.09	8.48	E-UK200_17	13	13	6308.25	135.77	2.12	110.58
E-UK20_18	2	1	1197.82	1.69	0.14	0.03	E-UK75_18	5	5	2460.94	49.78	1.98	6.02	E-UK200_18	14	13	5575.6	100.17	1.77	150.04
E-UK20_19	2	1	1080.58	11.66	1.06	0.03	E-UK75_19	5	5	2379.39	41.72	1.72	5.71	E-UK200_19	13	12	5013.21	108.26	2.10	197.80
E-UK20_20	2	1	1154.27	25.93	2.20	0.04	E-UK75_20	6	5	2711.73	44.30	1.60	4.97	E-UK200_20	13	13	6031.46	96.19	1.56	147.47

tackles a mixed fleet between EV and ICV, a full recharging policy, and realistic energy consumption, while GMFVRP-REC-PR possesses similar characteristics with two main differences: a partial recharging policy is utilized instead of a full recharging policy, and carbon emission minimization takes an important role as part of the objective function of this problem. Thus, we verify our ALNS algorithm by performing a comparison with Goeke and Schneider [30], which can be achieved by eliminating the emission cost from the objective function. We run our ALNS to solve each instance with five replications and we used the most minimum one to perform the comparison with Goeke and Schneider [30] since the benchmark results were also obtained from the best among the replications performed by Goeke and Schneider [30]. Although our algorithm is built to solve the partial recharging policy, it is still comparable since we adopt the dataset from Goeke and Schneider [30]. The gaps between our proposed ALNS and Goeke and Schneider [30] are calculated using Eq. (67). Table 4 presents a summary of the comparison.

$$Gap = \frac{Cost_{benchmark} - Cost_{ALNS}}{Cost_{benchmark}} \times 100\% \tag{67}$$

Table 7
Statistics for the destroy and repair operators.

Operator	Solution degradation without this operator	Average number of new best solutions found by the operator
Random Removal	−1.03	2.07
Worst Removal	−0.77	11.85
Route Removal	−0.05	17.11
Shaw Removal	−1.06	11.27
Station Vicinity Removal	−1.76	7.70
Sequential Insertion	−0.99	2.32
Sequential Insertion with Perturbation	−1.04	1.86
Regret Insertion	−1.39	23.08
Regret Insertion with Perturbation	−1.07	22.75

Table 4 summarizes the information of comparison. The columns ICV and EV represent the total utilized ICV and EV in all datasets, respectively, for each instance set. We take the average total cost of the twenty datasets in each instance set. These average values are listed in column *Average cost* in Table 4. Based on the conducted experiments, the largest average gap between our developed ALNS and that of Goeke and Schneider [30] is 1.192%, which can be found in *Instance set 7: 100 customers*. However, the proposed ALNS managed to obtain better results in Instance Set 9: 200 customers with the average gap of −0.988%. Although our ALNS algorithm does not outperform that of Goeke and Schneider [30], our algorithm could solve the problem with a partial recharging policy, which means closer to the real-world condition, while Goeke and Schneider [30] focused on solving the full recharging policy.

The average computational time of our proposed ALNS on solving EVRPTWMF ranges from 0.008 to 91.94 min. In particular, the ALNS performs faster compared to Goeke and Schneider [30] in solving instances with 10 to 50 customers, while it performs slower in solving the remaining instances. There are various factors influencing the computational time, i.e. CPU speed, memory size, operating system, compiler, computer program, and precision [54]. In addition, the dynamic programming procedure applied to EV routes is rather expensive in terms of computational time because it guarantees the optimal placement of recharging station(s) if a feasible route exists.

4.4. Performance of ALNS on GMFVRP-REC-PR instances

We now apply our ALNS algorithm to solve GMFVRP-REC-PR. For the small instances, we compare ALNS results with the results from CPLEX in Yu et al. [35]. CPLEX is run with a limit of two hours while ALNS is run with 5 replications on each instance. Table 5 presents the solution values and the computational times in seconds for two instance sets, 10 and 15 customers. Both Columns **ICV** and **EV** represent the number of utilized ICV and EV. Column **Total Cost** denotes the objective value of each instance obtained by CPLEX. In terms of computational time, our ALNS algorithm outperforms CPLEX solver significantly. For instances with 10 customers, our ALNS does not find an optimal solution for several instances, i.e. E-UK10_07, E-UK10_09, and E-UK10_10, with an average gap of 0.17%. However, we do obtain better solutions for several instances with 15 customers, i.e. E-UK15_06, E-UK15_13, and E-UK15_14, and one worse solution for one instance, i.e. E-UK15_04, with an average gap of −0.15%. CPLEX solved all small instances to the optimality with long computational times. Based on the conducted experiments, the ALNS performs under a less computational time with competitive solution quality compared to results from CPLEX solver. Consequently, our ALNS is the only method applied to remaining larger instances from Goeke and Schneider [30], varying from 20 customers to 200 customers.

Table 6 shows the results of all instances solved by ALNS. Column **Total Cost** presents the best objective value obtained among five replications. Columns $\mathbf{m_{IC}}$ and $\mathbf{m_E}$ denote the number of utilized ICVs and EVs, respectively. Two measurements of robustness are presented, i.e. standard deviation (**stdev**) and coefficient of variance (**CV**). As shown in Table 6, the computational time increases as the number of customers increases. Regarding to the robustness of the proposed ALNS, the results show that the standard deviation value tend to increase as the objective value increases. Therefore, CV is also presented as a companion to stdev. Based on the results, ALNS shows a relatively good stability since the average value of CV is 0.95%, while the maximum value is 2.76%.

$$\%\ Gap\ \ (cost) = \frac{(best\ total\ cost\ obtained\ by\ ALNS - Total\ cost) \times 100\%}{Total\ cost}$$

$$\%\ Gap\ \ (time) = \frac{(average\ computational\ time\ of\ ALNS - Time) \times 100\%}{Time}$$

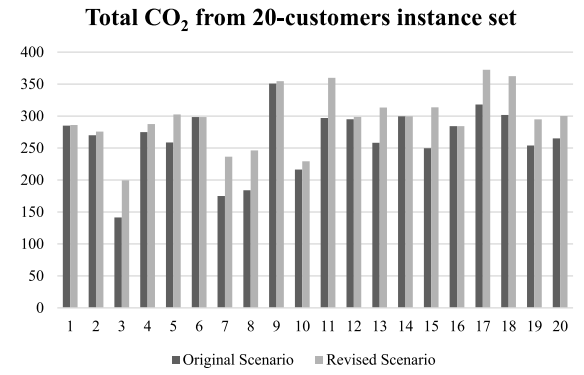
4.5. Sensitivity analyses on destroy and repair operators of ALNS

Table 7 summarizes the statistics on removing different destroy and repair operators. These statistics are built from five runs over a subset of instances. The first column shows the percentage of solution degradation when a particular operator is not considered. Based on this column, the station vicinity removal is the most useful operator among all the employed destroy operators, followed by the Shaw removal and random removal. Among all the repair operators, the regret insertion is superior to all other repair operators, followed by the regret insertion with perturbation. The success of regret insertions – both with and without perturbation – come from the criterion for selecting the node to be inserted. While the sequential insertions select the node with the least cost, the regret insertions consider not only the best position but also the 2nd and 3rd best positions in terms of cost. By harnessing this criterion, the regret insertions could help ALNS escape from the myopic behavior of sequential insertions. In the third column of Table 7, the percentage of new best solutions found by each operator is presented. Among all removal operators, the statistics show that route removal is the operator that finds the highest number of the best solutions. Again, the two versions of regret insertion become the superior ones in terms of the frequency of finding new best solutions.

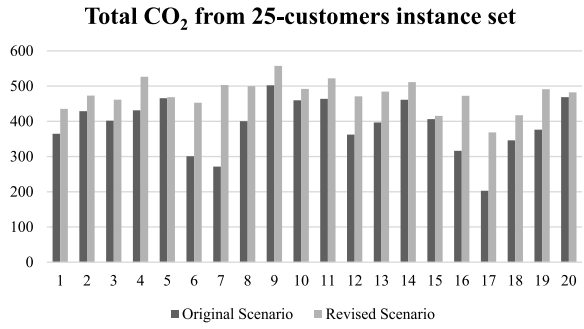
4.6. Potential emission reduction of solving GMFVRP-REC-PR

One of the main aims of this research is to show the potential emission reduction of addressing GMFVRP-REC-PR. In order to analyze it, we utilize the reduced version of GMFVRP-REC-PR by eliminating emission cost from the objective function. The original version of GMFVRP-REC-PR is stated as an original scenario, while the other version of the problem is named as a revised scenario. Four medium instance sets are selected for this purpose: 20, 25, 50, and 75 customers.

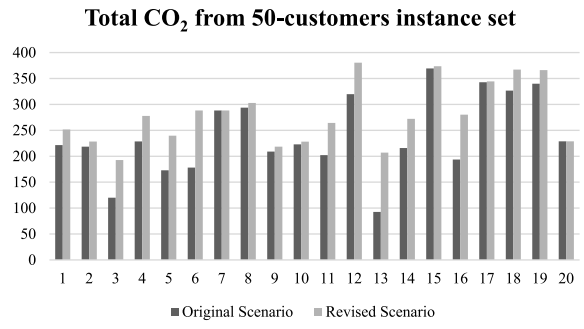
Our finding is that emission reduction could be gained by solving GMFVRP-REC-PR. Fig. 4(a) to (d) depict the difference



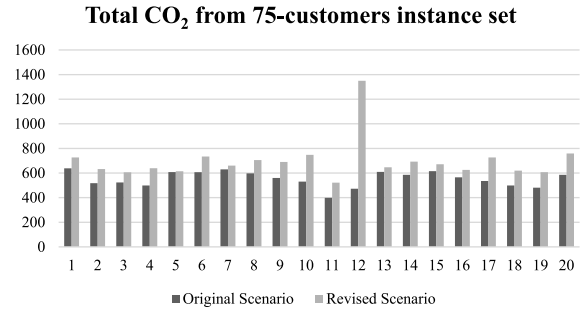
(a)



(b)



(c)



(d)

Fig. 4. Total emission cost calculated from the solutions in instance sets of (a) 20 customers, (b) 25 customers, (c) 50 customers, and (d) 75 customers.

in terms of total carbon emissions calculated from the obtained solution of each dataset in each instance set. The average CO_2 reductions are 11.09%, 17.95%, 15.68%, and 18.46% for instance sets with 20, 25, 50, and 75 customers, respectively. The reduction of CO_2 in solutions provided by the original scenario exists, because more customers are visited by EV fleets in the original scenario compared to the reduced scenario. Fig. 5 shows the average increase percentage of total distance traveled by the employed EV fleets.

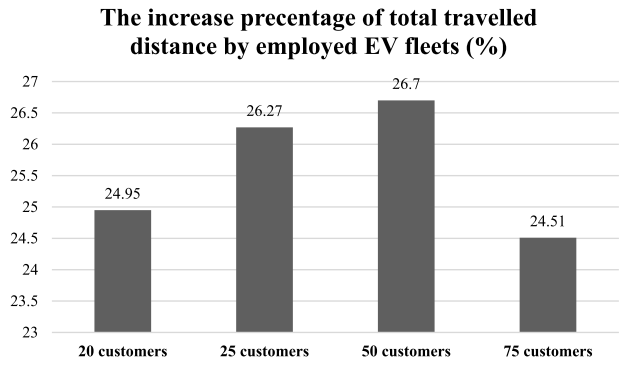


Fig. 5. The average increase percentage of total traveled distance by employed EV fleets resulting from the comparison between original and revised scenarios.

5. Conclusions

This research has solved a new variant of the routing problem, named the Green Mixed Fleet Vehicle Routing Problem with Realistic Energy Consumption and Partial Recharges. The experiments were first conducted on the EVRPTWMF, from which GMFVRP-REC-PR is extended. Although our proposed ALNS does not outperform the approach of Goeke and Schneider [30], it successfully obtains comparable results in terms of solution quality. The ALNS was then applied on GMFVRP-REC-PR instances. CPLEX solver were also employed to solve the small instances. Although both CPLEX and ALNS have a comparable performance in terms of solution's quality, CPLEX solver takes up significant high computational time while ALNS is able to perform much faster. Consequently, we only utilize our developed ALNS heuristic to deal with medium and large instance sets. Based on the numerical studies carried out during the experiment, potential carbon emission reduction could be achieved by solving GMFVRP-REC-PR.

Furthermore, future research can consider hybridizing other algorithms with ALNS to solve GMFVRP-REC-PR in order to achieve better results in terms of solutions quality and computational time. Another interesting extension is to reformulate the problem into a multi-objective problem and to develop an appropriate method to tackle the multi-objective version.

CRediT authorship contribution statement

Vincent F. Yu: Conceptualization, Methodology, Supervision, Resources, Writing - review & editing. **Panca Jodiawan:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Visualization, Writing - original draft. **Aldy Gunawan:** Methodology, Formal analysis, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was partially supported by the Ministry of Science and Technology of the Republic of China (Taiwan) under grant MOST 106-2410-H-011-002-MY3 and the Center for Cyber–Physical System Innovation from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

Appendix

See Table A.1.

Table A.1
Nomenclature table.

Symbol	Definition
P_M	Mechanical power requirement
m	Vehicle mass
a	Acceleration
c_d	The coefficient of aerodynamic drag
A	Vehicle frontal surface
v	Vehicle speed
g	Gravitational constant
α	The gradient angle of the road
c_r	The coefficient of rolling resistance
u_{ij}^k	Currently loaded amount of cargo that is brought by vehicle k from node i to node j
$m\left(u_{ij}^k\right)$	Total mass consisting of a vehicle's curb mass and u_{ij}^k
$p_{ij}^k\left(u_{ij}^k\right)$	Mechanical power requirement required by vehicle k to travel from node i to node j
Θ^d	Energy efficiency coefficient
φ^d	Discharging efficiency coefficient
$b_{ij}^k\left(u_{ij}^k\right)$	The electricity consumption of electric vehicle k to travel from node i to node j
t_{ij}	Traveling time from node i to node j
ξ	Fuel-to-air mass ratio
κ	Heating value of typical diesel fuel
k	Engine friction factor
N	Engine speed
D	Engine displacement
ψ	Fuel rate converting factor (gram per second to liters per second)
η	Efficiency parameter for diesel engines
η_{tf}	Drive train efficiency
$FR_{ij}^k\left(u_{ij}^k\right)$	The fuel consumption rate of internal combustion vehicle k while traveling from node i to node j
$f_{ij}^k\left(u_{ij}^k\right)$	The fuel consumption of internal combustion vehicle k while traveling from node i to node j
$0, N+1$	The origin and destination depot nodes
V	The customer nodes
F'	The set of dummy nodes of recharging stations
A	Set of arcs
d_{ij}	Distance between node i and j
t_{ij}	Traveling time between node i and j
c_t	Traveling cost for each kilometer traveled by the vehicle
q_i	Demand of node i
s_i	Service time of node i
$[e_i, l_i]$	Time windows of node i
K_E	Number of EVs
K_{IC}	Number of ICVs
Q	Capacity
B	Battery capacity
r	Recharging rate
FE	CO_2 emitted per liter of fuel
EM_{ij}	The total carbon emissions produced by an ICV while traveling from node i to node j
c_e	The carbon emission cost for each gram of emitted CO_2
$x_{ijk_E}^E$	1 if an EV k_E travels from node i to j $(k_E \in K_E; i, j \in 0 \cup V \cup F' \cup N+1)$
$x_{ijk_{IC}}^{IC}$	1 if an ICV k_{IC} travels from node i to j $(k_{IC} \in K_{IC}; i, j \in 0 \cup V \cup N+1)$
τ_i	arrival time at node i
$\tau_{N+1}^{k_E}$	arrival time of an EV k_E at node $N+1$
$\tau_{N+1}^{k_{IC}}$	arrival time of an ICV k_{IC} at node $N+1$
$u_0^{k_E}$	initial load brought by an EV k_E
$u_0^{k_{IC}}$	initial load brought by an ICV k_{IC}
$u_{ij}^{k_{IC}}$	amount of load brought by an ICV k_{IC} when traveling from node i to node j
$u_{ij}^{k_E}$	amount of load brought by an ICV k_E when traveling from node i to node j
$y_i^{K_E}$	remaining electric energy of an EV k_E upon arrival at node i
$Y_i^{K_E}$	amount of electric energy obtained by an EV k_E after recharging at recharging station i
$p_{ij}^{k_E}\left(u_{ij}^{k_E}\right)$	amount of mechanical power spent by an EV k_E when traveling from node i to node j and carrying a load of $u_{ij}^{k_E}$
$p_{ij}^{k_{IC}}\left(u_{ij}^{k_{IC}}\right)$	amount of mechanical power spent by an ICV k_{IC} when traveling from node i to node j and carrying a load of $u_{ij}^{k_{IC}}$
$b_{ij}^{k_E}\left(u_{ij}^{k_E}\right)$	amount of electric energy consumed by an EV k_E when traveling from node i to node j and carrying a load of $u_{ij}^{k_E}$
$f_{ij}^{k_{IC}}\left(u_{ij}^{k_{IC}}\right)$	amount of fuel consumed by an ICVk_{IC} when traveling from node i to node j and carrying a load of $u_{ij}^{k_{IC}}$
$\lambda_{gen}(\sigma)$	Generalized objective value of solution σ
γ_{cap}	Multiplication factor of capacity penalty
γ_{tw}	Multiplication factor of time windows penalty
γ_{batt}	Multiplication factor of battery capacity penalty
$Cap(\sigma)$	Capacity penalty of solution σ
$TW(\sigma)$	Time windows penalty of solution σ
$BT(\sigma)$	Battery capacity penalty of solution σ
$a_i^{\min}$	Earliest allowed arrival time at node i
$a_i^{\max}$	Arrival time at node i if as much fuel as possible has been recharged at preceding facilities
a_{ij}^{sl}	A possible slack between node i and node j that is happening due to time window limitations while traveling from node i to node j
a_i^{rt}	Inverse residual battery capacity at node i
a_{ij}^{add}	Additional fuel that has to be replenished at the preceding refueling facility to travel from node i to node j
$\tilde{a}_i^{\min}$	Adjusted value of $a_i^{\min}$ to prevent a repeated penalization
$\tilde{a}_i^{\max}$	Adjusted value of $a_i^{\max}$ to prevent a repeated penalization
$R(i,j)$	Relatedness between node i and node j
χ_d	Relatedness parameter on distance between node i and node j
χ_q	Relatedness parameter on demand difference between node i and node j
χ_e	Relatedness parameter on the difference of the earliest start of service between node i and node j
T_i^{cost}	Cost of path
T_i^{F}	Number of recharging station visits on path
T_i^{tMin}	Earliest arrival time on node i without running out of electric energy
T_i^{tMax}	Latest arrival time on node i recharging as much as possible at preceding recharging stations, without violating time windows
T_i^{rtMax}	Maximal amount of energy that can be recharged (expressed in time unit)
$P(D_i)$	The probability of choosing removal heuristic i
$P(R_i)$	The probability of choosing insertion heuristic i
π_i^D	The weight of removal heuristic i
π_i^R	The weight of insertion heuristic i
σ_i^D	The score of removal heuristic i
σ_i^R	The score of insertion heuristic i
$p(\sigma, \sigma', T)$	The acceptance probability of solution σ' by considering current solution σ and temperature T

References

[1] NASA, The causes of Climate Change, 2019, Available from: https://climate.nasa.gov/causes/.

[2] N. Zhang, P. Zhou, C.-C. Kung, Total-factor carbon emission performance of the chinese transportation industry: A bootstrapped non-radial malmquist index analysis, Renew. Sustain. Energy Rev. 41 (2015) 584–593.

[3] US Environmental Protection Agency, Fast facts on transportation greenhouse gas emissions, 2018, [cited 2019 13-02]; Available from: https://www.epa.gov/greenvehicles/fast-facts-transportation-greenhouse-gas-emissions.

[4] Eurostat, Greenhouse gas emission statistics - emission inventories, 2018, [cited 2019 2019.05.03]; Available from: https://ec.europa.eu/eurostat/statistics-explained/index.php/Greenhouse_gas_emission_statistics#Trends_in_greenhouse_gas_emissions.

[5] A. Ghorani-Azam, B. Riahi-Zanjani, M. Balali-Mood, Effects of air pollution on human health and practical measures for prevention in Iran, J. Res. Med. Sci.: Off. J. Isfahan Univ. Med. Sci. (2016) 21.

[6] A. Dean, D. Green, Climate change, air pollution and human health in Sydney, Australia: A review of the literature, Environ. Res. Lett. 13 (5) (2018) 053003.

[7] CNBC, Climate change to slow global economic growth, new study finds, 2019, Available from: https://www.cnbc.com/2019/08/20/climate-change-to-slow-global-economic-growth-new-study-finds.html.

[8] Center for Climate and Energy Solutions, U.S. state greenhouse gas emissions targets, 2019, [cited 2019 13-02]; Available from: https://www.c2es.org/document/greenhouse-gas-emissions-targets/.

[9] European Environment Agency, Greenhouse gas emissions from transport, 2018, [cited 2019 13-02]; Available from: https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-greenhouse-gases/transport-emissions-of-greenhouse-gases-11.

[10] A. Poullikkas, Sustainable options for electric vehicle technologies, Renew. Sustain. Energy Rev. 41 (2015) 1277–1287.

[11] L.C. Casals, et al., Sustainability analysis of the electric vehicle use in Europe for CO_2 emissions reduction, J. Clean. Prod. 127 (2016) 425–437.

[12] P. Jochem, S. Babrowski, W. Fichtner, Assessing CO_2 emissions of electric vehicles in Germany in 2030, Transp. Res. A 78 (2015) 68–83.

[13] M. Leonard, DHL brings electric delivery vehicle pilot to US, 2019, Available from: https://www.supplychaindive.com/news/dhl-electric-delivery-vehicle-pilot/568241/.

[14] N. Manthey, UPS converts 33 diesel vans to electric and hybrid, 2019, Available from: https://www.electrive.com/2019/10/10/ups-converts-33-diesel-vans-to-electric-and-hybrid/.

[15] FedEx, FedEx acquires 1, 000 chanje electric vehicles, 2018, Available from: https://about.van.fedex.com/newsroom/fedex-acquires-1000-chanje-electric-vehicles/.

[16] M. Coffman, P. Bernstein, S. Wee, Electric vehicles revisited: a review of factors that affect adoption, Transp. Rev. 37 (1) (2017) 79–93.

[17] P. Toth, D. Vigo, The Vehicle Routing Problem, SIAM, 2002.

[18] T. Vidal, G. Laporte, P. Matl, A concise guide to existing and emerging vehicle routing problem variants, European J. Oper. Res. (2019).

[19] M. Schneider, A. Stenger, D. Goeke, The electric vehicle-routing problem with time windows and recharging stations, Transp. Sci. 48 (4) (2014) 500–520.

[20] M. Keskin, B. Çatay, Partial recharge strategies for the electric vehicle routing problem with time windows, Transp. Res. C 65 (2016) 111–127.

[21] J. Martínez-Lao, et al., Electric vehicles in Spain: An overview of charging systems, Renew. Sustain. Energy Rev. 77 (2017) 970–983.

[22] Á. Felipe, et al., A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges, Transp. Res. E: Logist. Transp. Rev. 71 (2014) 111–128.

[23] M. Keskin, B. Çatay, A matheuristic method for the electric vehicle routing problem with time windows and fast chargers, Comput. Oper. Res. 100 (2018) 172–188.

[24] S. Pelletier, et al., Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models, Transp. Res. B 103 (2017) 158–187.

[25] T. Zündorf, Electric vehicle routing with realistic recharging models, 2014.

[26] A. Montoya, et al., The electric vehicle routing problem with nonlinear charging function, Transp. Res. B 103 (2017) 87–110.

[27] V.F. Yu, et al., A simulated annealing heuristic for the hybrid vehicle routing problem, Appl. Soft Comput. 53 (2017) 119–132.

[28] U. Breunig, et al., The electric two-echelon vehicle routing problem, Comput. Oper. Res. 103 (2019) 198–210.

[29] W. Jie, et al., The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology, European J. Oper. Res. 272 (3) (2019) 879–904.

[30] D. Goeke, M. Schneider, Routing a mixed fleet of electric and conventional vehicles, European J. Oper. Res. 245 (1) (2015) 81–99.

[31] R. Basso, et al., Energy consumption estimation integrated into the electric vehicle routing problem, Transp. Res. D 69 (2019) 141–167.

[32] G. Hiermann, et al., Routing a mix of conventional, plug-in hybrid, and electric vehicles, European J. Oper. Res. 272 (1) (2019) 235–248.

[33] G. Macrina, et al., The green mixed fleet vehicle routing problem with partial battery recharging and time windows, Comput. Oper. Res. 101 (2019) 183–199.

[34] G. Macrina, et al., An energy-efficient green-vehicle routing problem with mixed vehicle fleet, partial battery recharging and time windows, European J. Oper. Res. 276 (3) (2019) 971–982.

[35] V.F. Yu, et al., A mathematical programming model for the green mixed fleet vehicle routing problem with realistic energy consumption and partial recharges, in: 2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Macao, China (2019) 1339–1343.

[36] Y. Niu, et al., Optimizing the green open vehicle routing problem with time windows by minimizing comprehensive routing cost, J. Clean. Prod. 171 (2018) 962–971.

[37] R.A. Foroutan, J. Rezaeian, I. Mahdavi, Green vehicle routing and scheduling problem with heterogeneous fleet including reverse logistics in the form of collecting returned goods, Appl. Soft Comput. 94 (2020) 106462.

[38] Y.-J. Kwon, Y.-J. Choi, D.-H. Lee, Heterogeneous fixed fleet vehicle routing considering carbon emission, Transp. Res. D 23 (2013) 81–89.

[39] T. Bektaş, G. Laporte, The pollution-routing problem, Transp. Res. B 45 (8) (2011) 1232–1250.

[40] S. Ropke, D. Pisinger, An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, Transp. Sci. 40 (4) (2006) 455–472.

[41] A. Santini, An adaptive large neighbourhood search algorithm for the orienteering problem, Expert Syst. Appl. 123 (2019) 154–167.

[42] M.G. Avci, M. Avci, An adaptive large neighborhood search approach for multiple traveling repairman problem with profits, Comput. Oper. Res. 111 (2019) 367–385.

[43] R. Liu, Y. Tao, X. Xie, An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and synchronized visits, Comput. Oper. Res. 101 (2019) 250–262.

[44] M. Alinaghian, N. Shokouhi, Multi-depot multi-compartment vehicle routing problem, solved by a hybrid adaptive large neighborhood search, Omega 76 (2018) 85–99.

[45] R. Liu, Z. Jiang, A hybrid large-neighborhood search algorithm for the cumulative capacitated vehicle routing problem with time-window constraints, Appl. Soft Comput. 80 (2019) 18–30.

[46] M. Schiffer, G. Walther, An adaptive large neighborhood search for the location-routing problem with intra-route facilities, Transp. Sci. 52 (2) (2017) 331–352.

[47] Y. Nagata, O. Bräysy, W. Dullaert, A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows, Comput. Oper. Res. 37 (4) (2010) 724–737.

[48] V.C. Hemmelmayr, J.-F. Cordeau, T.G. Crainic, An adaptive large neighborhood search heuristic for Two-Echelon Vehicle Routing Problems arising in city logistics, Comput. Oper. Res. 39 (12) (2012) 3215–3228.

[49] P. Shaw, A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems, APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK, 1997.

[50] N. Labadie, et al., Metaheuristics for Vehicle Routing Problems, Wiley Online Library, 2016.

[51] G. Desaulniers, et al., Exact algorithms for electric vehicle-routing problems with time windows, Oper. Res. 64 (6) (2016) 1388–1405.

[52] T. Vidal, et al., A unified solution framework for multi-attribute vehicle routing problems, European J. Oper. Res. 234 (3) (2014) 658–673.

[53] E. Demir, T. Bektaş, G. Laporte, An adaptive large neighborhood search heuristic for the pollution-routing problem, European J. Oper. Res. 223 (2) (2012) 346–359.

[54] S.-W. Lin, V.F. Yu, Solving the team orienteering problem with time windows and mandatory visits by multi-start simulated annealing, Comput. Ind. Eng. 114 (2017) 195–205.