Electric Vehicle Routing Problem with Time Windows and Cargo Weight

Sina Rastani, Tuğçe Yüksel, and Bülent Çatay

Abstract There is growing interest in the utilization of electric vehicles (EVs) in logistics operations as they can cut dependency on fossil fuels, hence, significantly contribute to the efforts on reducing carbon emissions and air pollution. However, their limited driving range still remains as a major barrier in their adoption despite the advancements in battery technology. In this study, we extend the well-known Electric Vehicle Routing Problem with Time Windows by taking into account the cargo weight, which may play a crucial role in the operational efficiency of the EVs since it can affect the energy consumption significantly. We present the mixed-integer linear programming formulation of the problem and perform an extensive experimental study to investigate the influence of load on the routing decisions. We solve small-size instances using a commercial solver, and for the large-size instances, we develop a Large Neighbourhood Search algorithm. The results show that cargo weight may create substantial changes in the route plans and fleet size.

Keywords Vehicle routing · Electric vehicles · Time windows · Load · Energy consumption

S. Rastani (✉) · T. Yüksel · B. Çatay
Faculty of Engineering and Natural Sciences, Sabanci University, Istanbul, Turkey
e-mail: srastani@sabanciuniv.edu

T. Yüksel
e-mail: tugce.yuksel@sabanciuniv.edu

B. Çatay
e-mail: bulent.catay@sabanciuniv.edu

Smart Mobility and Logistics Lab, Sabanci University, Istanbul, Turkey

P. Golinska-Dawson et al. (eds.), *Smart and Sustainable Supply Chain and Logistics – Trends, Challenges, Methods and Best Practices*, EcoProduction,
https://doi.org/10.1007/978-3-030-61947-3_12

1 Introduction

Electric vehicles (EVs) have recently attracted great attention in transportation and logistics sector as they considerably reduce dependency to oil and consequently air pollution. The EVs are more efficient than conventional vehicles due to their electric motor and transmission system which transfers mechanical power to the wheels (Wu et al. 2015). Nevertheless, there are some technical drawbacks in case of using EVs. The most significant drawback is their battery capacity, which is low, and users should charge their batteries frequently in order to reach their destinations. Due to this restriction, routing an EV fleet has appeared as a challenging combinatorial optimization problem in the Vehicle Routing Problem (VRP) literature.

Electric Vehicle Routing Problem (EVRP) is an extension for VRP, where EVs are used in the fleet instead of fossil fuel vehicles. EVs reduce tailpipe emission and enhance green logistics. It tries to handle distribution tasks of logistics companies by minimizing the total energy consumption cost of serving customers and satisfying their demands. EVRP with Time Window (EVRPTW) is introduced by Schneider et al. (2014) where a full-recharge strategy was adopted. The authors developed the mathematical programming formulation of the problem and proposed a hybrid Variable Neighbourhood Search (VNS) and Tabu Search (TS) algorithm to solve it. Different variants of EVRP and EVRPTW were addressed in several studies including the cases of partial recharge (Bruglieri et al. 2015; Keskin and Çatay 2016), mixed fleet (Goeke and Schneider 2015; Hiermann et al. 2016), location routing (Schiffer and Walther 2017), fast charging (Felipe et al. 2014; Çatay and Keskin 2017; Keskin and Çatay 2018), non-linear charging function (Montoya et al. 2017; Froger et al. 2019), battery swapping (Yang and Sun 2015; Hof et al. 2017; Paz et al. 2018). Desaulniers et al. (2016) also studied EVRPTW and proposed a branch-price-and-cut algorithm to solve four different recharging strategies. Some recent studies have dealt with the availability of recharging stations and queueing for recharging service (Froger et al. 2017; Kullman et al. 2018; Keskin et al. 2019). A comprehensive review of the EV technology and survey of the EVRP variants may be found in (Pelletier et al. 2016; Pelletier et al. 2017; Erdelić and Carić 2019).

Energy consumption on the road does not only depend on the distance travelled but many other factors including the vehicle's weight, velocity, auxiliary equipment (internal factors) as well as ambient temperature and road gradient (external factors). These factors have been often neglected in the VRP literature either because they make the problem too complex to solve or the driving range is not an issue as the vehicles can easily refuel at a nearby gas station. However, they may play a critical for in the operational efficiency of the EVs since they can increase their energy consumption significantly (Rastani et al. 2019). Among them, the weight of the transported cargo may play a crucial role in route planning. The logistics operations of hypermarkets, hardware stores and other companies that deal with heavy loads are examples for which a load-dependent model produces more efficient transportation plans in comparison with basic routing models (Zachariadis et al. 2015), which constitutes the main motivation of this study.

Load Dependent Vehicle Routing Problem (LDVRP) was introduced in (Kara et al. 2007). They used the weighted distance objective and relate it with the energy requirements of vehicles. They proposed mathematical formulations for collection and distribution cases. Xiao et al. (2012) attacked the same problem by emphasizing the relation of the weighted distance with the fuel consumption of the vehicles within the context of Fuel Capacitated VRP. Zachariadis et al. (2015) extended LDVRP by considering simultaneous pick-ups and deliveries and proposed a local-search algorithm to solve large-scale instances.

In this study, we address the load-dependent variant of EVRPTW with partial recharges by taking into account the energy consumption associated with the cargo carried on the vehicle. We adopt a hierarchical objective function where the primary objective is to minimize the fleet size whereas the secondary objective is to minimize total energy consumption. We solve small-size instances using a commercial solver, and for the large-size instances, we develop a Large Neighbourhood Search (LNS) algorithm. The remainder of the chapter is organized as follows: Sect. 2 introduces the problem and formulates its mathematical programming model. Section 3 describes the proposed LNS method. Section 4 presents the experimental study and discusses the results. Finally, concluding remarks are provided in Sect. 5.

2 Problem Description and Mathematical Model

We tackle EVRPTW where a homogeneous fleet of EVs serve a set of customers with known demands, time windows, and service times. As opposed to previous studies in the literature which assume that the energy on the battery is consumed proportional to the distance traveled, we take into account the additional energy consumption related to freight load. Carrying more load by an EV causes more energy consumption. Furthermore, we allow partial recharging and its duration depends on the amount of energy transferred. Since it is a common practice in the real world to operate within the first phase of recharging where the energy transferred is a linear function of the recharge duration in order to prolong the battery life (Pelletier et al. 2017), we also assume a linear charging function. In addition, we assume that the EV can be recharged at most once between two consecutive customers, which is practical in last-mile logistics. We consider a pick-up problem where the load of the EV increases along its tour as it visits the customers. Each EV departs from the depot with full battery since it can be recharged overnight.

2.1 Mathematical Formulation

In line with the mathematical notation and modelling convention in the literature (Schneider et al. 2014; Keskin and Çatay 2016; Rastani et al. 2019) we define $V = \{1, \ldots, n\}$ as the set of customers and F as the set of recharging stations. Vertices 0

and $n+1$ denote the depot where each vehicle departs from 0 (departure depot) and returns to $n+1$ (arrival depot) at the end of its tour. We define $V_0 = V \cup \{0\}$, $V_{n+1} = V \cup \{n+1\}$ and $V_{0,n+1} = V \cup \{0, n+1\}$. Then, the problem can be represented on a complete directed graph $G = (N, A)$ with the set of arcs, $A = \{(i, j) \mid i, j \in N, i \neq j\}$ where $N = V_{0,n+1} \cup F$ is the total set of nodes on the network.

The energy consumption depends on the distance traveled and the total weight of the EV, which is affected by the cargo load carried on the EV. Each customer $i \in V$ has a positive demand q_i, service time s_i, and time window $[e_i, l_i]$. All EVs have a cargo capacity of C and a battery capacity of Q. At each recharging station, one unit of energy is transferred in g time units. The direct distance from node i to j is represented by d_{ij}.

Travel time from customer i to customer j is denoted by t_{ij} if the journey is direct and $\hat{t}_{ijs} = t_{is} + t_{sj} - t_{ij}$ is the additional travel time if it is via station s. Note that $\hat{t}_{ijs}$ does not include the recharging time at station s. The amount of extra energy needed in order to move one unit of cargo is represented by w. The total energy consumption starting from customer i to customer j is calculated as $(h + wu_i)d_{ij}$, where u_i is the weight of the load on the vehicle upon departure from customer i.

The decision variables ($y^k{}_i$, $\mathrm{y}^{\mathrm{k}}_{\mathrm{ijs}}$, and $Y^k{}_{ijs}$), keep track of battery SoC of vehicle k at arrival at customer/depot i, at arrival at station s on route (i, s, j), and at departure from station s on route (i, s, j), respectively. τ_i denotes the time when the loading starts at customer i. The binary decision variable $\mathrm{x}^{\mathrm{k}}_{\mathrm{ij}}$ takes value 1 if vehicle k travels from node i to node j, and 0 otherwise whereas the binary decision variable $\mathrm{z}^{\mathrm{k}}_{\mathrm{ijs}}$takes value 1 if vehicle k traverses arc (i, j), through station s.

$$\text{Minimize} \sum_{k \in K} (y_0^k - y_{n+1}^k) + \sum_{i \in V_0} \sum_{j \in V_{n+1}} \sum_{k \in K} \sum_{s \in F} (Y_{ijs}^k - y_{ijs}^k) \tag{1}$$

subject to

$$y_0^k = Q \;\; \forall k \in K \tag{2}$$

$$\sum_{\substack{j \in V_{n+1} \\ i \neq j}} \sum_{k \in K} x_{ij}^k = 1 \;\; \forall i \in V \tag{3}$$

$$\sum_{\substack{i \in V_0 \\ i \neq j}} x_{ij}^k - \sum_{\substack{i \in V_{n+1} \\ i \neq j}} x_{ji}^k = 0 \;\; \forall j \in V,\; k \in K \tag{4}$$

$$\sum_{s \in F} z_{ijs}^k \leq x_{ij}^k \quad \forall i \in V_0,\; j \in V_{n+1},\; k \in K,\; i \neq j \tag{5}$$

$$\tau_i + (t_{ij} + r_i)x_{ij}^k + \sum_{s \in F} (\hat{t}_{ijs} z_{ijs}^k + g(Y_{ijs}^k - y_{ijs}^k)) - l_0(1 - x_{ij}^k) \leq \tau_j$$
$$\forall i \in V_0,\, j \in V_{n+1}, k \in K, i \neq j \tag{6}$$

$$e_j \leq \tau_j \leq l_j \quad \forall j \in N \tag{7}$$

$$0 \leq y_j^k \leq y_i^k - (h + wu_i)d_{ij} + M(1 - x_{ij}^k + \sum_{s \in F} z_{ijs}^k) \quad \forall i \in V_0, j \in V_{n+1}, k \in K, i \neq j \tag{8}$$

$$y_j^k \leq Y_{ijs}^k - (h + wu_i)d_{sj} + M(1 - z_{ijs}^k) \quad \forall i \in V_0, j \in V_{n+1}, s \in F, k \in K, i \neq j \tag{9}$$

$$0 \leq y_{ijs}^k \leq y_i^k - (h + wu_i)d_{is} + M(1 - z_{ijs}^k) \quad \forall i \in V_0, j \in V_{n+1}, s \in F, k \in K, i \neq j \tag{10}$$

$$y_{ijs}^k \leq Y_{ijs}^k \leq Q z_{ijs}^k \quad \forall i \in V_0, j \in V_{n+1}, s \in F, k \in K, i \neq j \tag{11}$$

$$y_j^k \leq Q \sum_{i \in V_0} x_{ij}^k \quad \forall j \in V_{n+1}, k \in K \tag{12}$$

$$u_j \geq u_i + q_j \sum_{k \in K} x_{ij}^k - C(1 - \sum_{k \in K} x_{ij}^k) \quad \forall i \in V_0, j \in V_{n+1}, i \neq j \tag{13}$$

$$0 \leq u_i \leq c \quad \forall i \in V_{0,n+1} \tag{14}$$

$$x_{ij}^k \in \{0, 1\} \quad \forall i \in V_0, j \in V_{n+1}, k \in K \tag{15}$$

$$x_{ijs}^k \in \{0, 1\} \quad \forall i \in V_0, j \in V_{n+1}, s \in F, k \in K \tag{16}$$

The objective function (1) minimizes the total energy consumption. Constraints (2) set the initial battery SoC of EVs at departure to full. The connectivity of customer visits is imposed by constraints (3) whereas the flow conservation at each vertex is ensured by constraints (4). Constraints (5) make sure that vehicle k serves customer j after customer i if it travels from i to j by recharging its battery en-route. Constraints (6) guarantee the time feasibility of arcs emanating from the customers (the depot). Constraints (7) establish the service time windows restriction. Constraints (6) and (7) also eliminate the formation of sub-tours. Constraints (8)–(11) keep track of the battery SoC at each node and make sure that it never falls below zero where $M = Q + (h + w.\sum_{i \in V} q_i).max\{d_{ij}\}$. Constraints (8) establish the battery SoC consistency if the vehicle travels from customer i to customer j without recharging en-route. Constraints (9) determine battery SoC at the arrival at customer j if the vehicle

visits a recharging station after it has departed from customer i whereas constraints (10) check battery SoC at the arrival at a station if the battery is recharged en-route. Constraints (11) set the limits for battery SoC when the vehicle departs from a station. Constraints (12) allow positive battery SoC at the arrival of an EV at customer j only if that EV serves customer j. Constraints (13) keep track of the load of the vehicle throughout its journey. Constraints (14) ensure the non-negativity of the load on the vehicle and guarantee that the cargo capacity is not exceeded. Finally, constraints (15) and (16) define the binary decision variables.

2.2 Energy Consumption Function

The energy consumption of an EV that travels from one node to another depends on various factors such as its mass, shape, road gradient, acceleration, etc. By using tractive power requirements placed on the vehicle at the wheels, the power demand of a vehicle can be obtained using function (18) (Demir et al. 2012):

$$F = Ma + Mg\sin\theta + 0.5C_a\rho AV^2 + MgC_r\cos\theta \tag{17}$$

$$P_{tract}(kW) = Fv/1000 \tag{18}$$

where F shows the force function as calculated in (17), M is the total weight of the vehicle that consist of its curb weight and the cargo load (kg), a is the acceleration (m/s^2), g is the gravitational constant, θ is road gradient, C_d is the coefficient of aerodynamic drag, ρ is the air density in (kg/m^3), A is the frontal area, v is the speed (m/s), and C_r the coefficient of rolling resistance. The tractive power requirement can be converted to second-by-second engine power output (kW) as follows:

$$P = P_{tract}/\mu_{tf} + P_{acc} \tag{19}$$

where the vehicle's drive train efficiency is shown by μ_{tf} and P_{acc} is the power demand associated with the accessory equipment such as air conditioning, audio system and cabin lights, which is neglected in this study. Then, the energy consumption in (kWh/km) can be calculated as follows:

$$E = P/v. \tag{20}$$

3 Solution Methodology

We attempt to solve small-size instances using a commercial solver. To solve the large-size instances, we develop an LNS method. LNS was introduced by Shaw (1998) and aims at improving an initial solution by using several destroy and repair mechanisms iteratively. In each iteration, some customers are removed from the solution and reinserted into the routes to create a new feasible solution. This procedure is repeated for a predetermined number of iterations. LNS and Adaptive LNS (ALNS) have been successfully applied to many VRP variants including EVRPs and EVRPTWs (Keskin and Çatay 2016, 2018; Goeke and Schneider 2015; Hiermann et al. 2016; Schiffer and Walther 2017; Keskin et al. 2019; Wen et al. 2016; Schiffer et al. 2018).

We create the initial solution using the insertion heuristic in (Keskin and Çatay 2016) where the cost of inserting a customer into a route is calculated as $(h + wu_i)d_{ik}$ $+ (h + wu_k)d_{kj} - (h + wu_i)d_{ij}$. This insertion cost is calculated for all unvisited customers and the minimum cost insertion is performed by ensuring that the related constraints are not violated. If an EV runs out energy, a station may be inserted to make its tour energy feasible. We use First-Feasible Station Insertion algorithm which will be described in Sect. 3.3. If no customer can be feasibly inserted in the route, a new route is initialized, and the procedure is repeated until all customers are served.

Our LNS consist of customer removal and insertion mechanisms. In each iteration, a customer removal algorithm is applied on a feasible solution to remove a subset of customers from the routes. If any station is no longer needed in the partial solution, they are removed as well. Next, we apply a customer insertion algorithm that inserts all the customers removed to repair the solution in an attempt to obtain a new improved solution. Stations may be inserted to maintain the energy feasibility along the route. This procedure continues until the stopping criterion is satisfied, which is a limit on the number of iterations in our implementation. Note that the set of stations that can be visited between any two customers is reduced by using the dominance rules presented in (Bruglieri et al. 2016).

3.1 Customer Removal Operators

The current feasible solution is destroyed by removing γ customers. We use Worst-Consumption, Random Worst-Consumption, Shaw, Random Worst-Time, Random, Random Route Removal and Greedy Route Removal procedures of Keskin and Çatay (2016) by modifying them for the load dependent problem. The destroy operators are selected randomly.

- *Worst-Consumption* algorithm selects the customers with high energy consumption imposed to the route by visiting that customer, which is calculated as $(h +$

$wu_i)d_{ik} + (h + wu_k)d_{kj} - (h + wu_i)d_{ij}$ that considers distance and cargo load effect in energy consumption.

- *Random Worst-Consumption* sorts the customers with respect to the associated energy consumptions, considers a subset of $\sigma \times \gamma$ customers with highest costs to select γ customers randomly and remove them.
- *Shaw Removal* removes similar customers with respect to their energy consumption, earliest service time, being in the same route, and their demand. It randomly selects customer i and calculates the relatedness measure as $R_{ij} = \phi_1 h_i d_{ij} + \phi_2 |e_{i\_} e_j| + \phi_3 l_{ij} + \phi_4 |D_i - D_j|$ to find similar customers j. $\phi_1 - \phi_4$ are the Shaw parameters, $l_{ij} = -1$ if i and j are in the same route, 1 otherwise. Small R_{ij} shows high similarity. So, using the non-decreasing order of the relatedness value with customer i, γ customers are removed from the solution.
- *Random Worst-Time* algorithm is a version of Shaw Removal where ϕ_1, ϕ_3, ϕ_4 are set equal to 0. The customers are sorted in the non-decreasing order of their relatedness values and γ customers are randomly removed from the subset of $\sigma \times \gamma$ customers with lowest relatedness values.
- *Random Removal* mechanism randomly removes γ customers from the solution.
- *Random Route Removal* algorithm randomly removes ω routes from the solution.
- *Greedy Route Removal* mechanism sorts the routes in the non-decreasing order of the number of customers visited and removes ω routes which serve the least number of customers.

Note that the Route Removal algorithms attempt to reduce the fleet size.

3.2 Customer Insertion Operators

We adapt Random Greedy, Regret-2, Random Time-Based, Random Greedy with Noise Function, and Regret-2 with Noise Function repair algorithms in (Keskin and Çatay 2016; Demir et al. 2012) for our load-dependent case. In addition, we propose Exhaustive Greedy, Exhaustive Time-Based, Exhaustive Time-Based with Noise Function, and Random Time-Based with Noise Function mechanisms. The repair operators are selected randomly.

- *Random Greedy Insertion* selects a customer and inserts it in the best position which leads to least increase of energy consumption.
- *Regret-2 Insertion* try to avoid the higher costs in the subsequent iteration. It calculates the difference between the cost of the best insertion and the second-best insertion for all customers and selects the customer with the highest difference.
- *Random Time-Based Insertion* calculates insertion costs similar to the Exhaustive Time-Based algorithm, however, at first an unassigned customer is selected randomly, and the algorithm inserts it in its best position.
- *Random Greedy Insertion with Noise Function* is an extension of the Random Greedy Insertion mechanism with a degree of freedom. We use the same noise

function presented in (Demir et al. 2012). The cost of insertion using the freedom degree is calculated as $NewCost = ActualCost + \bar{d}\mu\epsilon$, where $\bar{d}$ represents the maximum distance in the network, the noise parameter used for diversification is shown by μ, and ϵ is a random number between [–1, 1].

- *Exhaustive Greedy Insertion* considers all possible insertion positions for all not-inserted customers and selects the customer-position matching which leads to least increase of energy consumption.
- *Exhaustive Time-Based Insertion* calculates the difference between the route duration after and before inserting a customer as the insertion cost. For all customers, the insertion costs in all possible positions are calculated and the customer with least insertion cost is selected.

Note that *Regret-2 with Noise Function, Exhaustive Time-Based with Noise Function, Random Time-Based with Noise Function* are extensions of Regret-2, Exhaustive Time-Based and Random Time-Based insertion mechanisms, respectively, using a similar noise function.

3.3 Station Removal and Insertion Operators

As we mentioned earlier, the unnecessary stations are removed from the partial solution obtained using the destroy operator. During the repair procedure, the insertion of a customer may not be feasible with respect to battery SoC. In that case, we first attempt to increase the recharge quantity if a station is visited prior to arriving to that customer. If the energy recharged at the station cannot be increased or no station is visited en-route we apply a station insertion operator to make the insertion feasible. We modified Best Station Insertion and Multiple Station Insertion operators from the literature (Keskin and Çatay 2016; Rastani et al. 2019) and applied them for the load dependent problem. Also, we develop First Feasible Station Insertion operator for this problem. Note that at most one station can be inserted between two consecutive customers in a route.

- *First-Feasible Station Insertion* considers the first customer (or depot) where the vehicle arrives at with negative SoC and checks the insertion of a station in the preceding arcs backwards. The first station which makes the problem feasible is inserted.
- *Best-Station Insertion* algorithm checks all possible stations in all possible arcs before the first customer (or depot) with negative SoC and inserts the best station in its best position.
- *Multiple-Station Insertion* algorithm inserts multiple stations into a route when the insertion of a single station cannot make the route feasible. A station is inserted on the arc traversed immediately before arriving at the customer (or depot) with a negative SoC where the vehicle is recharged up to the maximum level allowed by the battery capacity and time windows restrictions of the succeeding customers. If the SoC is still negative at that customer or if the vehicle runs out of energy before

reaching the inserted station, we attempt to insert another station prior to the last customer visited before traveling to the recently inserted station. This procedure is repeated until the route becomes energy feasible.

One of the First-Feasible Station Insertion and Best-Station Insertion algorithms is selected randomly. If it does not make the route feasible, we resort to Multiple-Station Insertion algorithm. Note that, we remove all stations in the solution after every β iterations and use Best-Station Insertion algorithm to insert stations to obtain an improved feasible solution.

4 Computational Study

We performed our computational tests using the dataset of Schneider et al. (2014) and Desaulniers et al. (2016) for the small-size and large-size instances, respectively. The small-size dataset consists of 36 instances involving 5, 10, and 15 customers, and the large-size dataset includes 56 instances generated based on the VRPTW instances of Solomon (1987). The instances are classified according to the geographic distribution of the customers: clustered (c-type), random (r-type), and half clustered half random (rc-type). Furthermore, in type-1 problems (i.e., subsets r1, c1, rc1) the planning horizon is shorter, and customers' time windows are narrower compared to type-2 problems (i.e., r2, c2, rc2). In our study, we only consider type-1 problems from the large-size dataset as they better exhibit the influence of recharging decisions on route planning (Keskin and Çatay 2016, 2018; Rastani et al. 2019).

In order to deal with realistic vehicle cargo capacity and customer demands, we assumed an electric truck based on the specifications provided in (Demir et al. 2012). Since capacity of this vehicle is 3650 kg, we converted the demand quantities to reasonable weights by multiplying each by (3650/original capacity) in order to observe the effect of cargo weight on energy consumption. We assumed a drive train efficiency of 0.9 as EVs are more efficient than internal combustion engine vehicles. Furthermore, since the EVs in the original data are assumed to consume one unit of energy per unit distance/time travelled, we used Eq. (20) to calculate the actual energy consumption of an empty vehicle (i.e., 6350 kg) per unit distance and scaled it to $h = 1$. We used the same approach to determine the energy consumption w associated with unit load carried. We consider a flat network where road gradients are zero and we neglected vehicle acceleration.

The small-size instances were solved using Gurobi 9.0 with a 2-hour time limit. LNS was employed to solve both small- and large-size instances. LNS was coded in Python 3.7.1 and all runs were performed on an Intel Core (TM) i7-8700 processor with 3.20 GHz speed and 32 GB RAM. We performed five runs for each instance. The number of LNS iterations is set to 15,000 for the small-size instances and 25,000 for the large-size.

The results for small-size instances are provided in Table 1. Column "Gurobi" shows the results using Gurobi and "LNS" provides the results obtained by the

Table 1 Results of small-size instances obtained using Gurobi and LNS

Instance	Gurobi						LNS			
	Load independent			Load dependent			Load dependent			
	#Veh	*EC*	*t* (s)	*#Veh*	*EC*	*t* (s)	*#Veh*	*EC*	*t* (s)	*Δ* (%)
r104c5-s3	2	137	<1	2	142	<1	2	142	11	0.00
r105c5-s3	2	156	<1	2	159	<1	2	159	8	0.00
r202c5-s3	1	129	<1	1	144	<1	1	144	16	0.00
r203c5-s4	1	179	<1	1	181	<1	1	181	9	0.00
c101c5-s3	2	258	<1	2	266	<1	2	266	9	0.00
c103c5-s2	1	175	<1	1	187	<1	1	187	10	0.00
c206c5-s4	1	243	<1	1	251	<1	1	251	10	0.00
c208c5-s3	1	164	<1	1	169	<1	1	169	9	0.00
rc105c5-s4	2	233	<1	2	257	<1	2	257	8	0.00
rc108c5-s4	2	254	<1	2	264	<1	2	264	10	0.00
rc204c5-s4	1	185	<1	1	189	<1	1	189	16	0.00
rc208c5-s3	1	168	<1	1	171	<1	1	171	14	0.00
r102c10-s4	3	249	<1	3	336	37	3	336	27	0.00
r103c10-s3	2	206	8	2	220	1507	2	220	29	0.00
r201c10-s4	1	242	<1	1	262	6	1	270	14	2.97
r203c10-s5	1	223	<1	1	227	7200	1	227	4	0.00
c101c10-s5	3	388	<1	3	410	117	3	410	23	0.00
c104c10-s4	2	274	<1	2	309	7200	2	308	48	– 0.26
c202c10-s5	1	304	<1	1	319	8	1	319	11	0.00
c205c10-s3	2	228	<1	2	234	148	2	234	26	0.00
rc102c10-s4	4	424	<1	**5**	475	435	5	475	17	0.00
rc108c10-s4	3	348	<1	3	365	4472	3	365	23	0.00
rc201c10-s4	1	413	<1	1	424	<1	2	327	28	–
rc205c10-s4	2	326	<1	2	335	432	2	335	25	0.00
r102c15-s8	5	413	3	5	431	7200	5	431	28	0.00
r105c15-s6	4	336	2	4	350	7200	4	350	32	0.00
r202c15-s6	1	507	594	**2**	365	7200	2	365	30	0.00
r209c15-s5	1	313	11	1	362	7200	1	360	25	– 0.60
c103c15-s5	3	348	33	**4**	393	7200	**3**	402	73	–
c106c15-s3	3	275	2	3	371	7200	3	352	52	– 5.27
c202c15-s5	2	384	11	2	408	7200	2	393	44	– 3.80
c208c15-s4	2	301	1	2	310	7200	2	310	47	0.00
rc103c15-s5	4	398	117	**5**	416	7200	**4**	416	45	–
rc108c15-s5	3	370	2002	**5**	453	7200	**3**	418	40	–
rc202c15-s5	2	394	1	2	403	7200	2	403	48	0.00
rc204c15-s7	1	382	7200	1	444	7200	1	446	9	0.45

proposed LNS algorithm. Column "Load Independent" reports the results for the case that does not consider the increased energy consumption associated with the cargo carried whereas column "Load Dependent" show the results for the case that considers the load on the vehicle. The comparison of these two columns exhibits the influence of the load on routing decisions. "#*Veh*", "*EC*", and "*t*" refer to the fleet size, energy consumption, and run time (in seconds), respectively. The results show that #*Veh* increases by one in four instances and by two in one instance (shown in bold). Notice that these increases are very significant considering the size of the fleet. Furthermore, we observe that *EC* values obtained in the load-independent case are far from the actual energy consumption found by taking into account the cargo load. Finally, we see that LNS finds (near-) optimal solutions in most of the instances while improving the solutions given by Gurobi in three instances with respect to #*Veh* and in four instances with respect to *EC*.

We solved the large-size instances for both load-independent and load-dependent cases using LNS. The results are provided in Table 2. We observe that #*Veh* increases by one in fourteen instances (shown in bold) in the load-dependent case compared to the load-independent. Furthermore, in the remaining 15 instances, *EC* increases by 14.3% on the average. These results show the importance of considering cargo weight in route optimization.

5 Conclusions and Future Research

In this study, we addressed EVRPTW with partial recharge by taking into account the energy consumption associated with the cargo carried on the vehicle. We formulated its 0–1 mixed integer linear programming model and used it to solve small-size instances. For solving the large-size instances, we proposed an LNS method. Our computational tests showed how the fleet size and/or energy consumption increase in comparison to the case where the load factor is neglected and revealed the importance of considering the weight of the vehicles for more accurate route planning. Future research on this topic may consider the road gradient as well. A loaded vehicle going uphill will consume significantly more energy. On the other hand, when it travels downhill it can recharge its battery with energy recuperation.

Table 2 Results of large-size instances obtained using LNS

Instance	Load independent			Load dependent		
	#Veh	*EC*	*t* (s)	*#Veh*	*EC*	*t* (s)
r101	19	1666	1397	19	1833	1323
r102	16	1491	1459	**17**	1746	1397
r103	14	1223	2365	14	1375	1910
r104	12	1115	3954	**13**	1253	2959
r105	15	1425	1460	**16**	1481	1690
r106	14	1298	2019	**15**	1469	2138
r107	12	1232	2482	**13**	1357	2211
r108	12	1082	3567	12	1190	3317
r109	14	1245	1952	14	1457	2188
r110	12	1151	3767	**13**	1256	3355
r111	12	1151	2749	**13**	1271	2955
r112	12	1091	4689	12	1204	3646
c101	12	1050	1740	12	1198	1460
c102	11	1206	2132	**12**	1210	1820
c103	11	1295	3231	11	1433	2418
c104	11	1049	5564	11	1456	3392
c105	11	1210	1696	**12**	1167	1829
c106	12	1034	2800	12	1171	3107
c107	12	1032	2860	12	1183	3116
c108	12	1087	3108	12	1215	3876
c109	11	1141	3905	**12**	1256	3987
rc101	17	1735	1785	17	1947	1612
rc102	15	1679	1911	**16**	1812	1653
rc103	14	1462	2363	14	1653	1985
rc104	12	1373	3547	**13**	1490	3340
rc105	15	1499	2151	15	1734	1712
rc106	14	1443	2331	**15**	1630	2136
rc107	13	1320	2924	13	1523	2339
rc108	12	1322	3730	**13**	1527	2634

Acknowledgments This study was partially supported by The Scientific and Technical Research Council of Turkey through Grant #118M412.

Appendix Parameters

The parameters used in LNS algorithm are displayed in Table 3.

Table 3 Parameter values

Param	Description	Value
γ	Number of customers removed	Random between [20%, 55%] of all customers
σ	Parameter used in Random Worst-Consumption and Random Worst-Time algorithm	1.5
ω	Number of routes removed	Random between [10%, 40%] of all routes
ϕ_1	First Shaw parameter	0.5
ϕ_2	Second Shaw parameter	0.25
ϕ_3	Third Shaw parameter	0.15
ϕ_4	Fourth Shaw parameter	0.25
μ	Noise parameter	0.1
ϵ	Random number for noise function	Random between [–1, 1]
α	First-Feasible Station Insertion selection probability	0.7
δ_1	Worst-Consumption selection probability	0.077
δ_2	Random Worst-Consumption selection probability	0.308
δ_3	Shaw selection probability	0.231
δ_4	Random Worst-Time selection probability	0.077
δ_5	Random selection probability	0.154
δ_6	Random Route Removal selection probability	0.077
δ_7	Greedy Route Removal selection probability	0.077
λ_1	Exhaustive Greedy Insertion selection probability	0.077
λ_2	Random Greedy Insertion selection probability	0.308
λ_3	Regret-2 Insertion selection probability	0.154
λ_4	Exhaustive Time-Based Insertion selection probability	0.077
λ_5	Random Time-Based Insertion selection probability	0.077

(continued)

Table 3 (continued)

Param	Description	Value
λ_6	Random Greedy with Noise Function Insertion selection probability	0.077
λ_7	Regret-2 with Noise Function insertion selection probability	0.077
λ_8	Exhaustive Time-Based with Noise Function insertion selection probability	0.077
λ_9	Random Time-Based with Noise Function insertion selection probability	0.077
β	Number of iterations to remove and reinsert stations	50

References

Bruglieri M, Mancini S, Pezzella F, Pisacane O (2016) A new mathematical programming model for the green vehicle routing problem. Electron Notes Discret Math 55:89–92. https://doi.org/10.1016/j.endm.2016.10.023

Bruglieri M, Pezzella F, Pisacane O, Suraci S (2015) A variable neighborhood search branching for the electric vehicle routing problem with time windows. Electron. Notes Discret. Math. 47:221–228. https://doi.org/10.1016/j.endm.2014.11.029

Çatay B, Keskin M 2017 The impact of quick charging stations on the route planning of electric vehicles. In: IEEE symposium on computers and communications (ISCC), 2017IEEE, vol 2017, pp 152–157. https://doi.org/10.1109/ISCC.2017.8024521.

Demir E, Bektaş T, Laporte G (2012) An adaptive large neighborhood search heuristic for the Pollution-Routing Problem. Eur J Oper Res 232(2):346–359. https://doi.org/10.1016/j.ejor.2012.06.044

Desaulniers G, Errico F, Irnich S, Schneider M (2016) Exact algorithms for electric vehicle-routing problems with time windows. Oper Res 64:1388–1405. https://doi.org/10.1287/opre.2016.1535

Erdelić T, Carić T (2019) A survey on the electric vehicle routing problem: variants and solution approaches. J Adv Transp. https://doi.org/10.1155/2019/5075671

Felipe Á, Ortuño MT, Righini G, Tirado G (2014) A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. Transp Res Part E Logist Transp Rev 71:111–128. https://doi.org/10.1016/j.tre.2014.09.003

Froger A, Mendoza JE, Jabali O, Laporte G (2019) Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. Comput Oper Res 104:256–294. https://doi.org/10.1016/j.cor.2018.12.013

Froger A, Mendoza JE, Jabali O, Laporte G (2017) A matheuristics for the electric vehicle routing problem with capacitated charging stations. CIRRELT.

Goeke D, Schneider M (2015) Routing a mixed fleet of electric and conventional vehicles. Eur J Oper Res 245:81–99. https://doi.org/10.1016/j.ejor.2015.01.049

Hiermann G, Puchinger J, Ropke S, Hartl RF (2016) The electric fleet size and mix vehicle routing problem with time windows and recharging stations. Eur J Oper Res 252:995–1018. https://doi.org/10.1016/j.ejor.2016.01.038

Hof J, Schneider M, Goeke D (2017) Solving the battery swap station location-routing problem with capacitated electric vehicles using an AVNS algorithm for vehicle-routing problems with intermediate stops. Transp Res Part B Methodol 97:102–112. https://doi.org/10.1016/j.trb.2016.11.009

Kara I, Kara BY, Yetis MK (2007) Energy minimizing vehicle routing problem. Combinatorial optimization and applications. Springer, Berlin Heidelberg, pp 62–71

Keskin M, Laporte G, Çatay B (2019) Electric vehicle routing problem with time-dependent waiting times at recharging stations. Comput Oper Res 107:77–94. https://doi.org/10.1016/j.cor.2019.02.014

Keskin M, Çatay B (2016) Partial recharge strategies for the electric vehicle routing problem with time windows. Transp Res Part C 65:111–127. https://doi.org/10.1016/j.trc.2016.01.013

Keskin M, Çatay B (2018) A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. Comput Oper Res 100:172–188. https://doi.org/10.1016/j.cor.2018.06.019

Kullman N, Goodson J, Mendoza JE (2018) 2018, June. Dynamic electric vehicle routing with mid-route recharging and uncertain availability, Seventh international workshop on freight transportation and logistics

Montoya A, Gueret C, Mendoza JE, Villegas JG (2017) The electric vehicle routing problem with nonlinear charging function. Transp Res Part B Methodol. 103:87–110. https://doi.org/10.1016/j.trb.2017.02.004

Paz JC, Granada-Echeverri M, Escobar JW (2018) The multi-depot electric vehicle location routing problem with time windows. Int J Ind Eng Comput 9:123–136. https://doi.org/10.5267/j.ijiec.2017.4.001

Pelletier S, Jabali O, Laporte G (2016) 50th anniversary invited article—goods distribution with electric vehicles: review and research perspectives. Transp Sci 50(1):3–22

Pelletier S, Jabali O, Laporte G, Veneroni M (2017) Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models. Transp Res Part B Methodol 103:158–187. https://doi.org/10.1016/j.trb.2017.01.020

Rastani S, Yüksel T, Çatay B (2019) Effects of ambient temperature on the route planning of electric freight vehicles. Transp Res Part D: Transp Environ 74:124–141

Schiffer M, Schneider M, Laporte G (2018) Designing sustainable mid-haul logistics networks with intra-route multi-resource facilities. Eur J Oper Res 265:517–532. https://doi.org/10.1016/j.ejor.2017.07.067

Schiffer M, Walther G (2017) The electric location routing problem with time windows and partial recharging. Eur J Oper Res 260(3):995–1013. https://doi.org/10.1016/j.ejor.2017.01.011

Schneider M, Stenger A, Goeke D (2014) The electric vehicle routing problem with time windows and recharging stations. Transp Sci 48:500–520. https://doi.org/10.1287/trsc.2013.0490

Shaw P (1998) Using constraint programming and local search methods to solve vehicle routing problems. In International conference on principles and practice of constraint programming. pp 417–431. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-49481-2_30

Solomon MM (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. Oper res 35(2):254–265. https://doi.org/10.1287/opre.35.2.254

Wen M, Linde E, Ropke S, Mirchandani P, Larsen A (2016) An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem. Comput Oper Res 76:73–83. https://doi.org/10.1016/j.cor.2016.06.013

Wu X, Freese D, Cabrera A, Kitch WA (2015) Electric vehicles' energy consumption measurement and estimation. Transp Res Part D 34:52–67. https://doi.org/10.1016/j.trd.2014.10.007

Xiao Y, Zhao Q, Kaku I, Xu Y (2012) Development of a fuel consumption optimization model for the capacitated vehicle routing problem. Comput Oper Res 39(7):1419–1431. https://doi.org/10.1016/j.cor.2011.08.013

Yang J, Sun H (2015) Battery swap station location-routing problem with capacitated electric vehicles. Comput Oper Res 55:217–232. https://doi.org/10.1016/j.cor.2014.07.003

Zachariadis EE, Tarantilis CD, Kiranoudis CT, (2015) The load-dependent vehicle routing problem and its pick-up and delivery extension. Transp Res Part B 71, 158–181. https://doi.org/10.1016/j.trb.2014.11.004