European Journal of Operational Research 289 (2021) 1153–1168

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing, Transportation and Logistics

Pickup and delivery problem with recharging for material handling systems utilising autonomous mobile robots

Sungbum Jun[a,*], Seokcheon Lee[b], Yuehwern Yih[b]

[a] Department of Industrial and System Engineering, Dongguk University, 3-26, Pil-dong 3ga, Chung-gu, Seoul, 100-715, Republic of Korea
[b] School of Industrial Engineering, Purdue University, 315 Grant St, West Lafayette, IN 47907, USA

ARTICLE INFO

Article history:
Received 10 June 2019
Accepted 23 July 2020
Available online 1 August 2020

Keywords:
Pickup and delivery problem
Autonomous mobile robots
Material handling
Memetic algorithm
Mixed-integer linear programming

ABSTRACT

Whereas automated guided vehicles (AGVs) have traditionally been used for material handling, the utilisation of autonomous mobile robots (AMRs) is growing quickly owing to their scalability, versatility, and lower costs. In this paper, we address the pickup and delivery problem with consideration of the characteristics of AMRs in manufacturing environments. To solve the problem, we first propose a new mathematical formulation with consideration of both partial and full recharging strategies for minimisation of the total tardiness of transportation requests. We then propose two constructive heuristic algorithms with high computation speed, which are called the Transportation-Request-Initiated Grouping Algorithm (TRIGA) and the Vehicle-Initiated Grouping Algorithm (VIGA). Additionally, we develop a memetic algorithm (MA) that incorporates a genetic algorithm into local-search techniques for finding near-optimal solutions within a reasonable time. We evaluate the performance of the proposed algorithms in comparison with two dispatching rules, genetic algorithm, and neighbourhood search through simulation experiments with three sets of problem instances under different battery levels. The simulation results indicate that the proposed algorithms outperform the others with regard to the average total tardiness and the relative deviation index.

1. Introduction

The role of material handling in a factory is the control of materials and products in the process of manufacturing as well as the incorporation of manual, semi-automated, and automated vehicles in support of production logistics. Traditionally, automated guided vehicles (AGVs) have been widely used for material handling in various manufacturing industries. These vehicles are guided by tape, wire, or magnetic tracks, and the distance between two locations is determined by predefined paths. Owing to their high investment costs and limited applications, AGVs have mainly been used in high-volume manufacturing operations.

While AGVs have played a significant role in production logistics, the utilisation of autonomous mobile robots (AMRs) for material handling is growing quickly as manufacturers are increasingly selecting semi-autonomous or fully autonomous vehicles owing to their scalability, versatility, and lower costs. According to the specifications of commercially available AGVs and AMRs used for material handling, the general characteristics of the two types of vehicles are summarised in Table 1. Also, detailed technical specifications for AGVs and AMRs in the market are compared in the Appendix.

AMRs have various advantages over traditional AGVs. First, owing to their autonomous ability, AMRs can easily avoid collisions and resolve conflicts between vehicles (such as deadlocks) by detouring slightly. Although AGVs can detect obstacles in front of them, they are not able to navigate around them, due to their fixed routes; thus, AGVs simply stop in their tracks until the tracks are free of obstacles. Unlike AGVs, AMRs can detour dynamically to avoid collisions based on data from built-in sensors such as cameras and laser scanners. Additionally, AMRs can be widely applied to perform additional tasks during transport, such as data collection in the era of industry 4.0, which features interconnected devices, dynamic reconfiguration, and mass data. Accordingly, even small- and medium-sized factories can exploit the advantages and potential of AMRs (Material Handling Institute, 2018).

Despite their great potential, the introduction of AMRs faces a variety of optimisation problems with regard to the determination of the best vehicle routes for serving of pickup and delivery operations in consideration of the load and battery capacities. These problems cause two challenges that are unique to AMRs.

The first challenge is the consideration of the new characteristics of AMRs. Owing to the small size of AMRs, their maximum payloads and travel times are highly restricted. In practice, fleets

* Corresponding author.
E-mail address: sbjun@dgu.ac.kr (S. Jun).

https://doi.org/10.1016/j.ejor.2020.07.049
0377-2217/

Table 1
Comparison between the AGV and the AMR.

	AGV	AMR
Navigation	Wire, magnet, reflective markers, radio-frequency identification (RFID), quick response (QR) code	Pre-loaded maps with laser and depth sensors for localisation
Speed	3 km/h	5–10 km/h
Payload	High (> 1500 kg)	Low (100–500 kg)
Collision avoidance	AGVs can only be routed while their path is free of obstacles.	AMRs can detour around obstacles and other AMRs dynamically.
Scalability	Low (new infrastructure and tracks must be installed)	High (each AMR can be controlled by a control system individually)
Power source	Large battery or inductive power transfer	Compact battery

of vehicles with different specifications can be used to increase the capacity and flexibility for material handling. Additionally, for minimisation of the travel time, the shortest paths in consideration of obstacles should be calculated before routing of vehicles. Finally, to fulfil customised orders of individual customers, factories rely on small production lot sizes and seek low levels of work-in-process inventory; thus, minimisation of delays for transportation requests should be considered in the process of mass customisation (Bechtsis, Tsolakis, Vlachos & Srai, 2018; Berman, 2002). Therefore, it is crucial to identify new characteristics, such as pathfinding, maximum payloads and battery levels, and to design new solution approaches to address these issues.

The second challenge facing AMR deployment is computational complexity. While mathematical models such as mixed-integer linear programming (MILP) can guarantee the optimal solution for small-sized problems, they are not applicable to larger problems. When the problem size (such as the number of vehicles or transportation requests) is scaled up, extremely long computational time is incurred, due to the nature of NP-hard problems. Thus, there is a significant need for efficient algorithms that can find a good solution within a reasonable time in various environments.

The goal of this study is to address the optimisation problem for the minimisation of the total tardiness under limited load and battery capacities and to develop algorithms for effective solutions. The reminder of paper is organised as follows. In the next section, previous research on vehicle routing problems (VRPs) for material handling is reviewed. A new mathematical model for the pickup and delivery problem (PDP) with partial recharging is presented in Section 3. In Section 4, new solution approaches for the defined problem are proposed. Section 5 summarises the results of experiments for verification of the proposed approaches. Finally, Section 6 presents the conclusions and identifies areas of future research.

2. Literature review

The vehicle routing problem with AMRs is a combinatorial optimisation problem entailing identification of a set of optimal routes (with consideration of recharging) whereby a fleet of vehicles can serve transportation requests. Notably, this problem is closely related to the PDP, which concerns parts or finished products that must be collected from pickup locations and delivered to paired delivery locations. This section provides a detailed overview of the research streams focusing on PDPs and VRPs with recharging (see Table 2).

Owing to the need for optimal solutions to the PDP, exact approaches such as mathematical models have been researched. Dumas, Desrosiers and Soumis (1991) proposed an exact algorithm with a mathematical formulation for the PDP with time windows (PDP-TW) wherein the objective is the minimisation of the sum of the total travel cost. Savelsbergh and Sol (1995) formulated a mathematical model for the general PDP. Additionally, they considered a dynamic version of the model, because most PDP situations, being demand-responsive, require that the solution should be re-optimised at a specific point in order to consider newly arriving requests. Li et al. (2015) designed branch-and-cut and branch-and-price algorithms for the PDP with a loading cost. Mahmoudi and Zhou (2016) proposed a dynamic-programming-based approach to find optimal solutions for the PDP-TW with a single vehicle.

However, because of the NP-hard nature of the PDP, exact approaches are only applicable to small-sized problems. Thus, heuristic and metaheuristic algorithms are widely used to obtain good solutions within a reasonable time. Mitrović-Minić, Krishnamurti and Laporte (2004) proposed double-horizon-based heuristics for the dynamic PDP-TW, in which future transportation requests cannot be stochastically modelled or predicted. Ganesh and Narendran (2007) considered a PDP having a mix of nodes (pure delivery, pure pickup or both delivery and pickup). To solve the problem, they designed a multi-phase constructive heuristic called CLOVES (Cluster and orient Nodes and Assign Vehicles) that clusters nodes based on proximity and orients them with a genetic algorithm (GA) for an intensive final search.

Dispatching rules (DRs) are widely used in practice, particularly in the case of PDPs for AGVs, owing to their short computation times. Ho and Chien (2006) addressed the control problem of multiple-load AGVs and proposed DRs for two sub-problems: task-determination and delivery-dispatching. Additionally, they performed simulation experiments to elucidate the mutual effects and identify the best combination of rules. Ho and Liu (2009) addressed the performance of pickup-DRs and load-selection rules for multiple-load AGVs by using a detailed flow chart of the multiple-load AGV control process.

Regarding metaheuristic algorithms, evolutionary algorithms such as neighbourhood search (NS), GA, particle swarm optimisation (PSO), and memetic algorithm (MA) are widely used to solve various PDPs. Li, Chen and Prins (2016) proposed an adaptive large NS (ALNS) approach for the PDP with time windows, profits, and reserved requests. The proposed ALNS involves a local search procedure with destroy and repair operators, which can adapt the selection probabilities of operators according to the statistics of successive segments. Muñoz-Carpintero, Sáez, Cortés and Núñez (2015) proposed a methodology based on generic evolutionary algorithms, which considers different configurations of PSO and the GA within a proposed ad-hoc methodology to solve the dynamic PDP, where quick and efficient real-time solutions are needed. Zhu, Xiao, He, Ji and Sun (2016) addressed the one-to-many-to-one dynamic PDP by proposing a multi-objective MA based on locality-sensitive hashing that utilises the synergy of the multi-objective evolutionary algorithm and local search.

In addition to the PDPs, the limited driving range of electric vehicles (EVs) as well as necessary charging times has to be considered in order to realise electric mobility concepts in real-world applications. To deal with the limited battery capacity of EVs, various recharging strategies have been proposed. In general, these strategies can be categorised as full recharging, partial recharging, and battery swapping.

Table 2
Overview of research related to PDPs and VRPs with recharging.

Reference	Problem type	Recharging
Lin et al. (2016)	VRP	Full Recharging
Schneider et al. (2014)	VRP-TW	Full Recharging
Hiermann et al. (2016)	VRP-TW	Full Recharging
Macrina et al. (2019)	VRP-TW	Partial Recharging
Keskin and Çatay (2016)	VRP-TW	Partial Recharging
Bruglieri et al. (2017)	VRP-TW	Partial Recharging
Keskin and Çatay (2018)	VRP-TW	Partial Recharging with Fast Chargers
Schiffer and Walther (2017)	LRP-TW	Partial Recharging
Wen et al. (2016)	VSP	Partial Recharging
Masmoudi et al. (2018)	DARP	Battery Swapping
Muñoz-Carpintero et al. (2015)	PDP	No Recharging
Savelsbergh and Sol (1995)	PDP	No Recharging
Ho and Chien (2006)	PDP	No Recharging
Ho and Liu (2009)	PDP	No Recharging
Chen et al. (2016)	PDP	Full Recharging
Mitrović-Minić et al. (2004)	PDP-TW	No Recharging
Dumas et al. (1991)	PDP-TW	No Recharging
Li and Lim (2001)	PDP-TW	No Recharging
Mahmoudi et al. (2016)	PDP-TW	No Recharging
Li et al. (2016)	PDP-TW	No Recharging
Wang and Cheu (2013)	PDP-TW	Full Recharging
Zhu et al. (2016)	1-M-1 PDP	No Recharging
Dragomir et al. (2018)	Multi-Depot PDP	No Recharging

The full-recharging strategy recharges vehicles to the full battery capacity for each recharging stop and thus, partial recharging is prevented. Lin et al. (2016) investigated an electric VRP that finds the optimal routing strategy for limited-range electric vehicles that might have to recharge at a charging station during their operations. In the proposed mathematical model, the battery level always goes back to full capacity when leaving a charging station. Schneider, Stenger and Goeke (2014) and Hiermann, Puchinger, Ropke and Hartl (2016) addressed an electric VRP with time windows (VRP-TW) and recharging stations that recharge vehicles at the maximum energy capacity. In the case of PDPs with EVs, full recharging strategies also are widely used to deal with the issue of the limited running time of vehicles (Chen, Zhang, Pourbabak, Kavousi-Fard & Su, 2016; Wang & Cheu, 2013).

Partial recharging, on the other hand, enables vehicles to recharge only as much energy as required to finish the next trip. By minimising charging times, the partial recharging can improve system performance but at the cost of increased computational complexity. Recently, VRP-TW with partial recharging constraints have been considered by various researchers (Keskin and Çatay 2016; Bruglieri, Mancini, Pezzella, Pisacane & Suraci, 2017; Macrina, Laporte, Guerriero & Pugliese, 2019). In addition to VRP-TW, variants of VRPs such as the location routing problem with time windows (LRP-TW) and the vehicle scheduling problem (VSP) with partial recharging have been presented (Schiffer & Walther, 2017; Wen, Linde, Ropke, Mirchandani & Larsen, 2016). Especially, in order to investigate the benefits of fast recharges, Keskin and Çatay (2018) modelled the electric VRP-TW for allowance of partial recharges with three recharging configurations referred to as normal, fast and super-fast recharges.

In the case of battery swaps, the discharged battery is replaced with a full one at each visit to a battery swapping station, and the time for swapping batteries is generally shorter than other recharging methods. Masmoudi, Hosny, Demir, Genikomsakis and Cheikhrouhou (2018) addressed the Dial-a-Ride Problem (DARP) with EVs and battery swapping stations; DARP assigns a fleet of EVs to serve a set of transport requests and EVs can be recharged by swapping their batteries for charged ones at any of the stations.

While considerable work related to various extensions of the PDP has been performed, the PDP with AMRs has not been extensively studied. It is worthy of further research for the following reasons. First, to deal with rush orders and work-in-process products, consideration of due-date-based objective functions and their minimisation is crucial. Additionally, several constraints related to limited payloads and battery levels should be considered. Especially, partial charging strategies should be considered in order to identify the optimal solution as well as the benefits from reduced charging times. Finally, the performance of the algorithms should be evaluated via experiments in concert with path-finding. For these reasons, there is an absolute necessity for solution approaches that can find efficient routes within a reasonable time with consideration of the distinguishing characteristics of AMRs.

Thus, in the following section, we first formulate a mathematical model for a PDP with partial recharging operations for AMRs in order to identify the optimal solution under the best charging scenario. We then propose new heuristic and metaheuristic algorithms and evaluate their performance in comparison with other well-known heuristic algorithms.

3. Problem definition

The PDP has a set of transportation requests with two paired nodes (pickup and delivery) to be processed by any vehicle not exceeding the maximum payload. In addition to these basic assumptions regarding the PDP, we consider constraints related to recharging, such as detours for visiting recharging stations and recharging times.

The following assumptions are made in this study:

- Each transportation request—having a due date and weight—consists of two nodes: a node for picking up work-in-process parts from one work centre and a node for delivering them to another work centre. The pickup and delivery nodes are only served at designated loading/unloading stations in each work centre (the yellow and red areas in Fig. 1 represent the work centres and their stations, respectively).

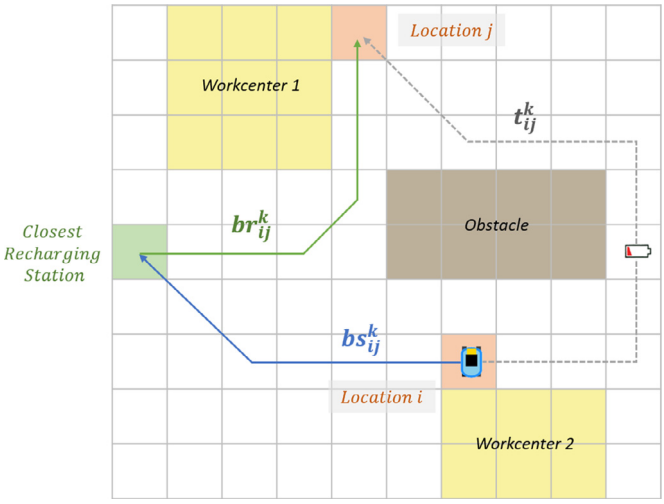


Fig. 1. Detours required for recharging.

- Each AMR has a maximum payload, initial and maximum travel times, and a speed.
- The objective function is applied to minimise the total tardiness, which is defined as the sum of all the tardiness values of the transportation requests. The tardiness is defined as max(completion time of delivery – due date, 0).
- The shortest routes and travel times (considering obstacles) between all possible locations, including the initial locations of all vehicles and loading/unloading stations, are precalculated using path-finding algorithms. The brown areas in Fig. 1 represent obstacles such as pillars and other machines.
- While an AMR is travelling, the battery level decreases linearly with the distance; therefore, the AMR might need to visit the closest charging station between the pickup and delivery locations. If there are more than two charging stations, the closest one is chosen and its capacity (the maximum number of AMRs to be charged) is assumed as unlimited. In our mathematical model, to find the optimal solution under the best charging scenario, the battery can be charged to any level. However, in the case of heuristic algorithms, only full recharging is considered because it is difficult to determine the exact amount of battery capacity to be recharged for upcoming transportation requests. The charging time depends on the recharging rate and the amount to be charged.
- The required detours for charging are precalculated. As shown in Fig. 1, these detours can be divided into two routes: the blue route from the origin i to the charging station (bs_{ij}^k), and the green route from the charging station to the destination j (br_{ij}^k).

To formulate a PDP with partial recharging for minimisation of the total tardiness, the following indices, sets, parameters, and decision variables are defined.

Indices and Sets

$i,\ j$	indices of transportation requests
n	number of transportation requests
k	index of vehicle, $k \in V$
P	set of pickup locations, $P = \{1, 2, \ldots, n\}$
D	set of delivery locations, $D = \{n+1, n+2, \ldots, 2n\}$
N	set of all locations for vehicles, $N = \{0,\ 2n+1\} \cup P \cup D$, where 0 represents the initial location and $2n+1$ represents the destination for each vehicle

Parameters

q_i	weight to be added at each location (positive for pickup and negative for delivery)
w^k	maximum load capacity of vehicle k
d_i	due date of transportation request i ($d_i = d_{i+n},\ \forall i \in P$)
s_i^k	service time of vehicle at location i
t_{ij}^k	required travel time from location i to location j for vehicle k
rs^k	recharging rate of vehicle k
e^k	initial energy capacity of vehicle k in travel time
m^k	maximum energy capacity of vehicle k in travel time
l^k	minimum energy capacity of vehicle k in travel time
bs_{ij}^k	detouring time from location i to the closest charging station for vehicle k
br_{ij}^k	detouring time from the closest charging station to location j for vehicle k

Decision Variables

X_{ij}^k	1 if vehicle k travels from location i to location j; 0 otherwise
Y_{ij}^k	1 if vehicle k requires recharging for travelling from location i to location j; 0 otherwise
C_i^k	elapsed time when vehicle k arrives at location i
L_i^k	completion time of vehicle k when leaving location $i+n$
Q_i^k	load capacity of vehicle k when leaving location i
E_i^k	current energy capacity of vehicle k in travel time after leaving location i
R_{ij}^k	energy capacity of vehicle k in travel time to be recharged when leaving the closest station between location i and location j
T_i	tardiness of location i

The proposed mathematical model is as follows:

$$Min \quad \sum_i T_i$$

$$s.t. \quad X_{ii}^k = 0\ \forall i \in N, \quad \forall k \in V \tag{1}$$

$$X_{i+n,i}^k = 0\ \forall i \in P, \quad \forall k \in V \tag{2}$$

$$X_{0,i+n}^k = 0\ \forall i \in P, \quad \forall k \in V \tag{3}$$

$$\sum_{k \in V} \sum_{j \in N} X_{ij}^k = 1 \quad \forall i \in P \tag{4}$$

$$\sum_{j \in N} X_{ij}^k - \sum_{j \in N} X_{i+n,j}^k = 0\ \forall i \in P, \quad \forall k \in V \tag{5}$$

$$\sum_{j \in N} X_{0,j}^k = 1 \quad \forall k \in V \tag{6}$$

$$\sum_{i \in P} X_{i+n,2n+1}^k = 1 \quad \forall k \in V \tag{7}$$

$$\sum_{j \in N} X_{ji}^k - \sum_{j' \in N} X_{ij'}^k = 0 \quad \forall i \in P \cup D, \quad \forall k \in V \tag{8}$$

$$C_j^k \geq C_i^k + s_i^k + t_{ij}^k - M\left(1 - X_{ij}^k\right) + Y_{ij}^k\left(bs_{ij}^k + br_{ij}^k - t_{ij}^k\right) + R_{ij}^k / rs^k \quad \forall i, j \in N, \quad \forall k \in V \tag{9}$$

$$C_{i+n}^k \geq C_i^k\ \forall i \in P, \quad \forall k \in V \tag{10}$$

$$Q_i^k \geq q_i \quad \forall i \in N, \quad \forall k \in V \tag{11}$$

$$Q_i^k \le w^k \quad \forall i \in N, \quad \forall k \in V \tag{12}$$

$$Q_i^k \le w^k + q_i \quad \forall i \in N, \quad \forall k \in V \tag{13}$$

$$Q_j^k \ge Q_i^k + q_j - M\left(1 - X_{ij}^k\right) \quad \forall i, j \in N, \quad \forall k \in V \tag{14}$$

$$E_0^k \ge e^k \quad \forall k \in V \tag{15}$$

$$E_i^k \le m^k \quad \forall i \in N, \quad \forall k \in V \tag{16}$$

$$R_{ij}^k \le M\left(Y_{ij}^k\right) \quad \forall i, j \in N, \quad \forall k \in V \tag{17}$$

$$R_{ij}^k \le m^k - \left(E_i^k - bs_{ij}^k\right) \quad \forall i, j \in N, \quad \forall k \in V \tag{18}$$

$$E_i^k \ge bs_{ij}^k\left(Y_{ij}^k\right) + l^k \quad \forall i, j \in N, \quad \forall k \in V \tag{19}$$

$$E_i^k + R_{ij}^k - t_{ik}^k - Y_{ij}^k\left(bs_{ij}^k + br_{ij}^k - t_{ik}^k\right) \ge E_j^k - M\left(1 - X_{ij}^k\right) \quad \forall i, j \in N, \quad \forall k \in V \tag{20}$$

$$L_i^k \le M \sum_{j \in N} X_{j,i+n}^k \quad \forall i \in P, \quad \forall k \in V \tag{21}$$

$$L_i^k \le C_{i+n}^k \quad \forall i \in P, \quad \forall k \in V \tag{22}$$

$$L_i^k \ge C_{i+n}^k - M\left(1 - \sum_{j \in N} X_{j,i+n}^k\right) \quad \forall i \in P, \quad \forall k \in V \tag{23}$$

$$T_i \ge \sum_{k \in V} L_i^k - d_i \quad \forall i \in P \tag{24}$$

and

$$X_{ij}^k \in \{0, 1\} \quad \forall i, j \in N, \quad \forall k \in V$$

$$Y_{ij}^k \in \{0, 1\} \quad \forall i, j \in N, \quad \forall k \in V$$

$$R_{ij}^k \ge 0 \quad \forall i, j \in N, \quad \forall k \in V$$

$$C_i^k \ge 0,\ L_i^k \ge 0,\ E_i^k \ge 0,\ Q_i^k \ge 0 \ \ \forall i \in N, \quad \forall k \in V$$

$$T_i \ge 0 \quad \forall i \in N$$

In this formulation, the objective is to minimise the tardiness of delivery for all transportation requests. Constraints (1)–(3) eliminate infeasible movements of vehicles according to X_{ij}^k. Constraint (1) indicates that a vehicle cannot go around at the same location. Constraints (2) and (3) prevent a vehicle from travelling to a delivery location without picking up products.

Constraints (4) and (5) indicate that only one vehicle can pick up the material and that it should be delivered by the same vehicle, respectively. Constraints (6) and (7) make the vehicles start from their origin and return to their destination after delivering the last request, respectively. Constraint (8) represents the conservation of incoming and outgoing flows at location i.

Constraint (9) determines the completion times of pickup and delivery according to X_{ij}^k, with consideration of the service and recharging times. Constraint (10) ensures that a delivery operation cannot start before its pickup operation. Constraints (11)–(14) describe the capacity constraints for vehicles in consideration of the maximum weight and accumulated weights according to X_{ij}^k. They ensure that the vehicle load never exceeds the maximum payload of the vehicle. Constraints (11)–(13) impose the feasible range of load capacity ($max\{0,\ q_i\} \le Q_i^k \le min\{w^k, w^k + q_i\}$). Constraint (14) linearises a non-linear constraint ($Q_j^k \ge (Q_i^k + q_i)X_{ij}^k$) for consideration of added or reduced weights according to pickup or delivery operations by introducing a large number M. Such constraints are very similar to those for the dial-a-ride problem (Cordeau, 2006).

Constraints (15)–(20) guarantee that the remaining travel time is within the range of the minimum travel time (l^k) to the maximum travel time (m^k) and that recharging is required if the remaining travel time is shorter than the required travel time for movement to the next location. l^k stores the minimum energy to cope with unexpected changes and prevent vehicles from being discharged during operations. m^k and e^k represent the maximum energy capacity for each vehicle and the initial energy capacity at the beginning, respectively.

Constraint (19) prevents AMRs from being fully discharged after finishing all tasks in the optimal solution. Constraints (21)–(23) convert nonlinear constraints into linear ones via the Big-M method with L_i^k, and Constraint (24) determines the tardiness of all requests. In considering partial recharges, R_{ij}^k determines the energy capacity to be recharged for each recharging stop between locations i and j while keeping the minimum energy level that the vehicle needs to finish its next trip.

In order to consider full recharging strategy as well, the proposed MILP with partial recharging can be converted to one with full recharging by adding Constraint (25), which recharges vehicles to the full battery capacity for each recharging stop.

$$R_{ij}^k \ge m^k - \left(E_i^k - bs_{ij}^k\right) - M\left(1 - Y_{ij}^k\right) \quad \forall i, j \in N, \quad \forall k \in V \tag{25}$$

4. Solution approaches

In this section, we propose new algorithms (and some widely used ones), for solving the problem defined in Section 3. Two heuristic algorithms and a new evolutionary algorithm are first described, and then the other well-known heuristic algorithms are explained.

4.1. Constructive heuristic algorithms

To respond rapidly to various changes in manufacturing environments, constructive heuristic algorithms with high computation speeds are essential. We propose two constructive heuristic algorithms based on grouping strategies to solve the defined problem efficiently. The notations used in Algorithms 1 and 2 are explained in Table 3.

4.1.1. Transportation-request-initiated grouping algorithm (TRIGA)

In the TRIGA, a transportation request first selects a vehicle and then chooses other transportation requests by sacrificing its tardiness on a certain level (*Threshold*). This grouping strategy determines the set of transportation requests from a collaborative perspective and identifies the best sequence among them. The grouping procedures of TRIGA are illustrated in Fig. 2.

Grouping_Tardy represents the increased tardiness incurred when the currently selected vehicle serves the candidate request with other chosen requests, while *Alternative_Tardy* is the tardiness when the other vehicle serves the candidate request. *Threshold* represents the acceptable level for the increase of tardiness (or the expense of the tardiness for the global improvement) when

Algorithm 1: TRIGA.

```
obtain a list of requests, a list of vehicles, Grouping_Size, and Threshold
do {
    P_Set = Ø, D_Set = Ø
    initialise SR and SV
    SR ← a request with the earliest due date in a list of unserved requests
    add SR's pickup and delivery nodes to P_Set and D_Set, respectively
    SV ← a vehicle with the earliest expected_arrival_time(each vehicle, pickup node of SR)
    do {
        initialise CR and CV
        CR ← a request by the lowest slack value (candidate's due date – SR's due date – travel time between SR and candidate's pickup node)
        CV ← a vehicle with the earliest expected_arrival_time(each vehicle, pickup node of CR)
        add CR's pickup and delivery nodes to P_Set and D_Set, respectively
        Alternative_Tardy ← sum of tardiness when CV serves CR and SV serves SR
        Grouping_Tardy ← minimum sum of tardiness when SV serves P_Set and D_Set, considering all possible sequences
        if (Grouping_Tardy – Alternative_Tardy > Threshold) {remove CR's pickup and delivery nodes to P_Set and D_Set, respectively}
    } while (P_Set ≤ Grouping_Size)
    mark all requests added to P_Set and D_Set as served by SV
     update Total_Tardiness and SV's current location and completion time after serving
} while (unserved requests exist)
Function expected_arrival_time(Vehicle, Node)
    if (Vehicle's battery is not enough to serve Node) {
        return travel time from Vehicle's current position to the closest station + charging time to be full + travel time from the closest station to Node's
          location
    } else {return travel time from Vehicle's current position to Node's location}
End Function
```

Algorithm 2: VIGA.

```
obtain a list of requests, a list of vehicles, and Grouping_Size
do {
    initialise SR and SV
    SV ← a vehicle with the lowest sum of expected_arrival_time(each vehicle, pickup node of each request) for all unserved requests
    P_Set = Ø, D_Set = Ø
    do {
        calculate rank values of SV for all unserved requests based on expected_arrival_time(each vehicle, pickup node of each request)
        add pickup and delivery nodes of a request having the lowest rank to P_Set and D_Set, respectively
    } while (P_Set ≤ Grouping_Size)
    mark all requests added to P_Set and D_Set as served by SV and update Total_Tardiness
} while (unserved requests exist)
```

Table 3
Notations used in Algorithms 1 and 2.

Notation	Definition
SR	Selected transportation request to be processed next
SV	Selected AMR to serve transportation requests
CR	Candidate transportation request to be grouped with *SR*
CV	Candidate AMR to check the expected performance when it serves *CR*
P_Set	Set of pickup nodes
D_Set	Set of delivery nodes
expected_arrival_time	Expected arrival time considering the time to be taken for recharging
Alternative_Tardy	Expected tardiness when *CV* serves *CR* instead of grouping
Grouping_Tardy	Expected tardiness when *SV* serves *CR* instead of *CV*
Threshold	Threshold for grouping transportation requests
Grouping_Size	Maximum number of transportation requests to be grouped
Total_Tardiness	Total tardiness

deciding the appropriate size of grouped requests. By comparing *Grouping_Tardy* with *Alternative_Tardy*, the selected AMR (*SV*) decide whether the candidate request is grouped in P_Set and D_Set with consideration of *Threshold* and all possible sequences of nodes. To find the best solution among different parameters (*Grouping_Size* and *Threshold*), the TRIGA evaluates solutions by changing the *Grouping_Size* and *Threshold* values by *Threshold_Step* and returns the best solution.

The detailed procedures of the TRIGA are presented in Algorithm 1.

4.1.2. Vehicle-Initiated grouping algorithm (VIGA)

The basic idea of the VIGA is that an idle vehicle selects one or more transportation requests from a list of requests according to grouping decisions. As shown in Fig. 2, the selected AMR (*SV*) groups a set of pickup nodes up to *Grouping_Size* based on the expected arrival times and delivers them together to minimise the travelling times and tardiness. Similarly to the TRIGA, the VIGA evaluates solutions for all *Grouping_Size* values and returns the best solution. The main steps of the VIGA are presented in Algorithm 2.

4.2. Memetic algorithm

The MA incorporates local-search techniques into a GA to prevent premature convergence and to improve the solution quality by balancing exploration and exploitation in its search mechanism. Our proposed MA consists of two phases: local search and the GA. The local search is applied as a local exploitation of a given seed

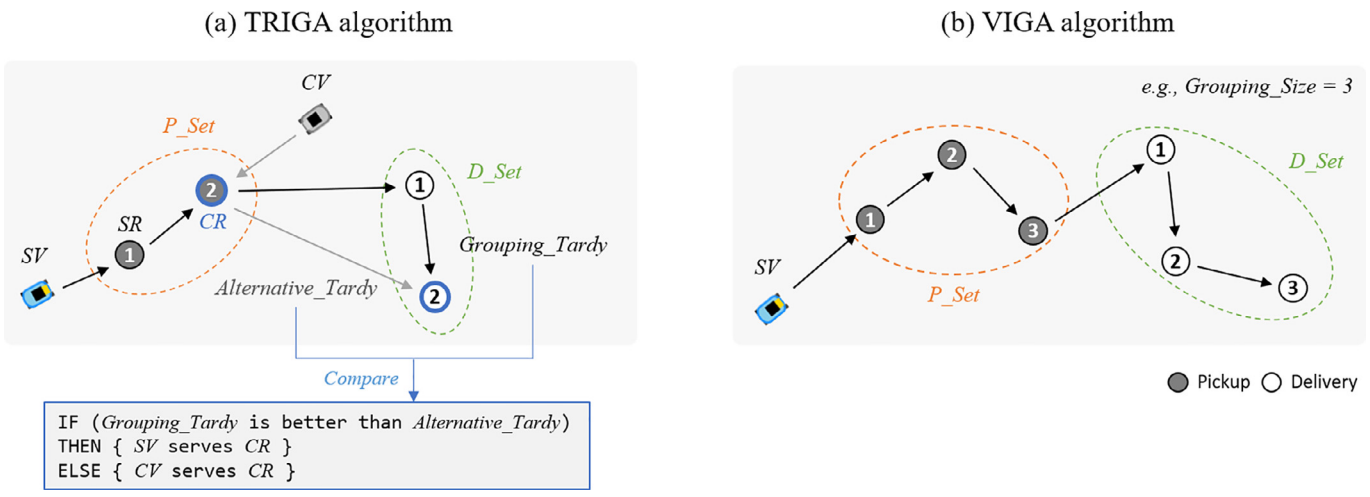


Fig. 2. Illustration of grouping procedures of TRIGA and VIGA.

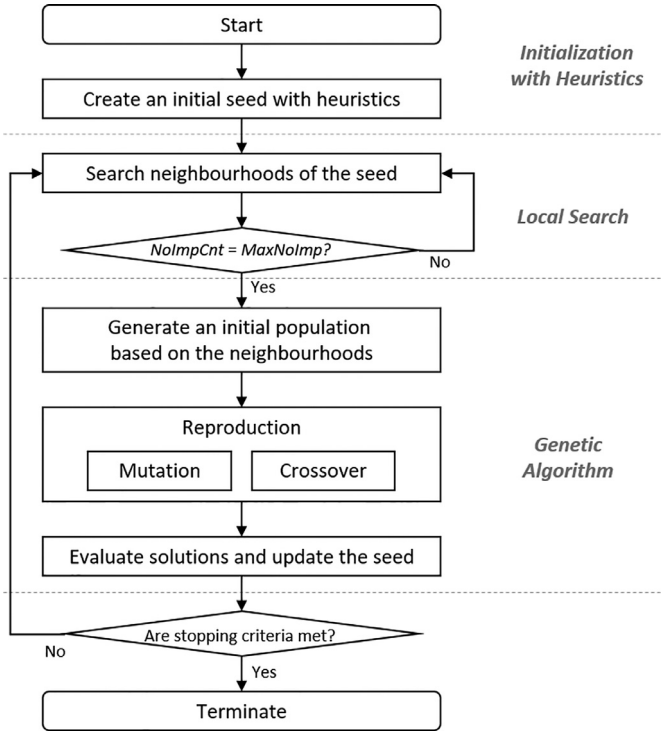


Fig. 3. Overall MA framework.

solution from heuristics, and the GA is used for global exploration. The entire process of the MA is summarised in Fig. 3.

4.2.1. Chromosome representation

Each chromosome consists of two vectors: a node-sequencing vector and a vehicle-assignment vector, as shown in Fig. 4. The total length of each vector is equal to the total number of pickup and delivery nodes of all requests ($2n$). The numbers in the node-sequencing vector represent the indices of the nodes. Similarly, the number in the vehicle-assignment vector represents the assigned vehicle index. The structure of a chromosome is shown in the figure. The node-sequencing vector in a chromosome is structured such that the first entry of a particular value (e.g., 1) would represent the pickup node and the next entry of the same value in the vector would represent the corresponding delivery node. In the chromosome-decoding phase, the assigned pickup and delivery nodes and their sequence are retrieved from a chromosome for each vehicle, and the completion time and tardiness of each transportation request is calculated with consideration of the required recharging. Similar to the algorithms presented in Section 4.1, the recharging policy is as follows: vehicles with insufficient battery charge to serve the next assigned request are fully recharged.

4.2.2. Local search phase

Before searching for better solutions locally, a seed solution is created according to the best solution among the solutions using the heuristics in Section 4.1. On the basis of the seed solution, a set of neighbourhoods is generated from the intra-route and inter-route exchanges. The local search algorithm is implemented using two local-search operators: intra-route relocation and inter-route relocation, which are widely used in various local-search techniques for the PDP (Li & Lim, 2003; Li et al., 2016).

Intra-Route relocation. The locations of the two pickup or delivery nodes in a chromosome are exchanged for the same vehicle. Fig. 5 shows an intra-route relocation whereby a delivery node of request 2 and a pickup node of request 5 are relocated.

Inter-Route relocation. The two pickup and delivery nodes are removed from a vehicle and are inserted into another vehicle. Fig. 6 illustrates an inter-route relocation whereby the pickup and delivery nodes of request 5 are inserted into another vehicle.

4.2.3. Genetic algorithm phase

In the GA phase, mutation and crossover operations can maintain diversity in a population and allow the local-search phase to avoid local minima by preventing solutions from becoming too similar.

Generation of initial population. The initial population is generated from the combination of randomly generated chromosomes and the given number of neighbourhoods in the previous phase.

Reproduction. To change the vehicle assignment part in the selected chromosome, a vehicle mutation operator called vehicle-based mutation selects a number of operations (denoted as *mv_num*) randomly and reassigns them to another vehicle. Regarding operation mutation, the swap-node mutation method, whereby two genes are selected randomly and their positions are exchanged, is applied, and the operation is repeated *ms_num* times (Tan, Lee, Zhu & Ou, 2001).

In the case of crossover, a two-point crossover based on non-wrapping order crossover (NWOX) is applied (Cicirello, 2006). NWOX strongly preserves the relative order as well as the absolute positions within the parent permutations. An example of the implementation of the machine crossover operator is shown in Fig. 7.

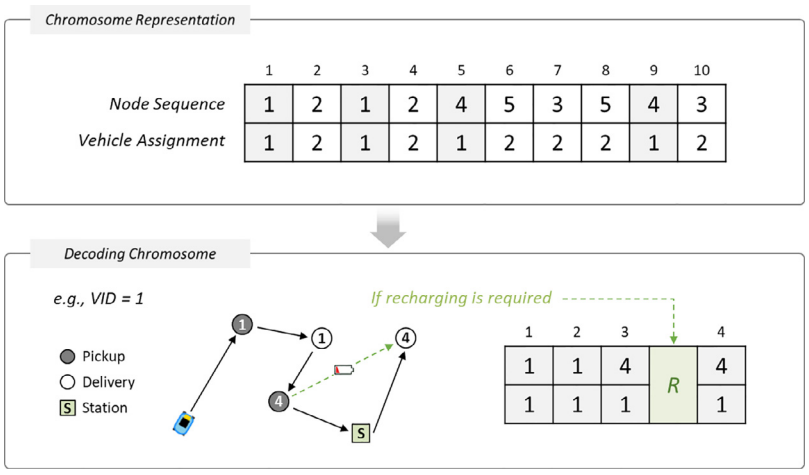


Fig. 4. Chromosome representation and decoding procedure with consideration of recharging.

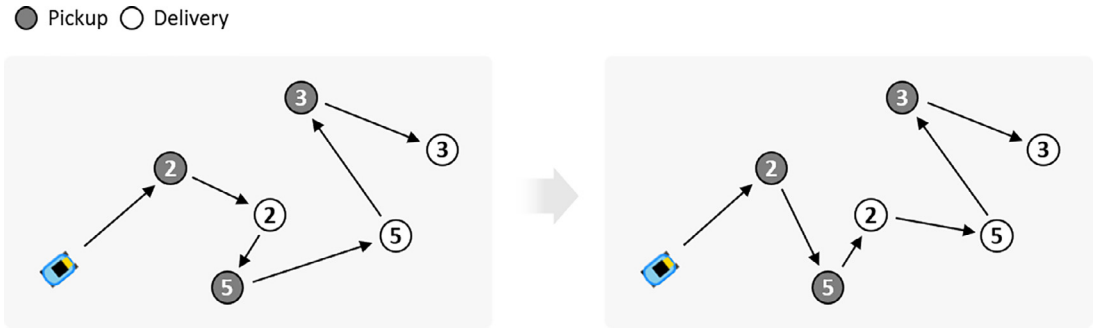


Fig. 5. Intra-route relocation.

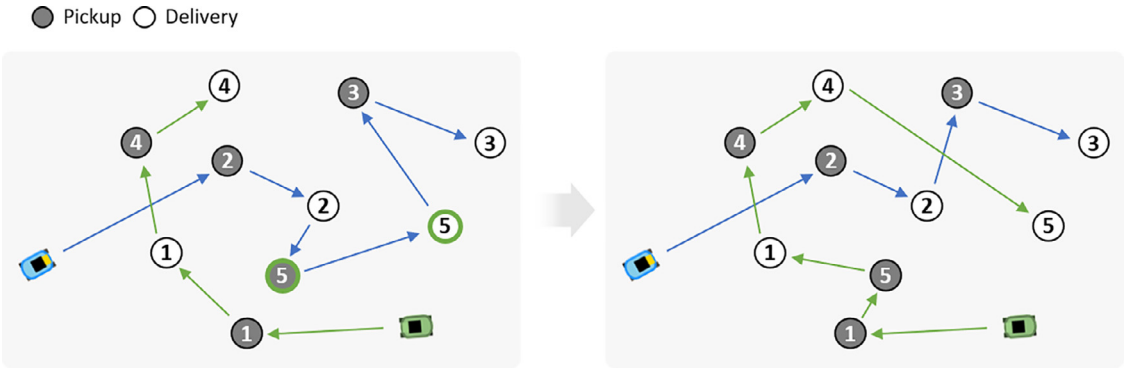


Fig. 6. Inter-route relocation.

Evaluation and updating of seed. After the generation of chromosomes according to the given number of populations, the chromosomes are evaluated by the total tardiness, and to break ties, the total distances are used. The seed solution from the previous local search phase is replaced with the best chromosome in the population.

4.2.4. Termination

The termination criterion is the maximum computation time.

4.3. Dispatching rules (DR)

In this study, two widely used DRs based on two criteria—distance (Shortest Travel Distance Rule; STD) and due dates (Earliest Due Time; EDT)—were implemented (Ho & Chien, 2006). The STD is based on the Nearest Vehicle rule proposed by Egbelu and Tanchoco (1984), which has been shown to perform well for throughput maximisation. Meanwhile, the EDT calculates the rank of transportation requests according to the due dates when a vehicle is idle; then, the transportation request having the earliest due date is selected.

4.4. Neighbourhood search (NS)

A NS algorithm based on two local-search operators—Swapping Pairs Between Routes (SBR) and Within-Route Insertion (WRI)—from the literature (Curtois, Landa-Silva, Qu & Laesanklang, 2018; Nanry & Barnes, 2000) was implemented. The termination criterion was the maximum computation time.

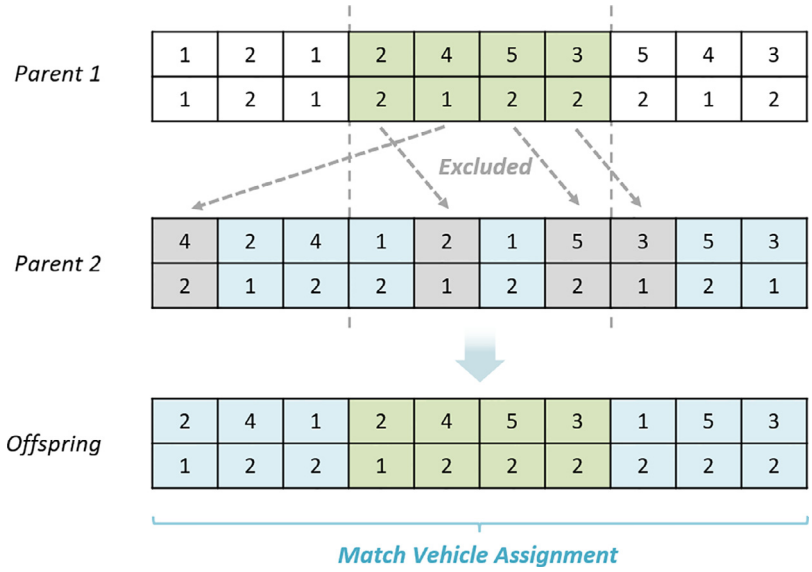


Fig. 7. Illustration of the crossover operation.

Table 4
Parameters for the generation of problem instances.

Parameter	Value
Map width (meters)	1000
Map height (meters)	800
Grid size	1×1 m^2
Number of work centres	20
Range of width and depth for work centres (meters)	[50, 100]
Number of obstacles	30
Range of width and depth for obstacles (meters)	[10, 50]
Neighbourhood type of A*	Moore
Heuristic function of A*	Euclidean
Range of due dates (hours)	[0.1, 0.3]
Range of weights for transportation requests (kg)	[1, 8]
Service time (hours)	0.05
Vehicle speed (km/h)	4
Recharge time from empty to full (hours)	3
Maximum payload (kg)	100
Maximum travel time (hours)	9

4.5. Simple genetic algorithm (SGA)

The simple genetic algorithm (SGA was implemented by using the same chromosome encoding, mutation, and crossover operators that were discussed in Section 4.2.3. The next surviving chromosomes were selected using the tournament selection method, and the maximum computation time was used as the stopping criterion. The detailed SGA parameters are presented in Table 4.

5. Experimental results

5.1. Experimental design

The algorithms proposed in Section 4 were tested via simulation experiments. The design of the experiments is summarised in Fig. 8. After layouts were randomly generated, the shortest paths between loading or unloading stations and recharging stations (with consideration of obstacles) were determined using the A* algorithm, which is widely used (Wang et al. 2015; Rodenberg, Verbree & Zlatanova, 2016). The detailed procedures of the A* algorithm are presented in Appendix A. All paths were precalculated and given to solution approaches as input parameters (t_{ij}^k, bs_{ij}^k, and br_{ij}^k) in order to minimise the total tardiness of the transportation requests.

To evaluate and compare the performances of the proposed mathematical model and the other algorithms, three problem sets were designed. Each problem set had 30 problem instances with different layouts. Each problem instance contained a set of transportation requests and AMRs on a map with the work centres, stations, obstacles, and shortest routes precalculated by A*.

In order to find the best parameters for meta-heuristic algorithms, we performed sensitivity analysis on parameters as shown in Fig. 9. Experimental tests with different configurations based on a problem instance of 50 requests with 10 AMRs were conducted by changing the parameter values (number of population, mutation and crossover rates) for both the GA and MA. The range of population is set from 100 to 300 at an interval of 100 and mutation and crossover rates are also set from 0.25 to 0.45 at an interval of 0.1 based on parameters from the previous literature for VRPs and PDPs (Chand, Mishra & Dehuri, 2010; Ting, Liao, Huang & Liaw, 2017; Utamima, Pradina, Dini & Studiawan, 2015).

The initial population for the GA and MA was generated by adding solutions using the heuristics and DRs in Sections 4.1 and 4.3, as well as randomly generated solutions for the population diversity. All of the algorithms were coded in C# and run on an Intel Xeon E5–1620 3.6 GHz processor with 16 GB RAM. For MILP with two recharging strategies (full and partial), IBM CPLEX optimiser version 12.7.1 with the default settings was used, and the run-time was limited to 3600 s. The maximum computation time of the meta-heuristic algorithms was determined to be 5 min for prompt reflection of shop floor changes. Tables 4 and 5 present the data relevant to the experimental design parameters used in this study for the algorithms.

5.2. Results and discussion

To evaluate the average performances for the algorithms in various recharging scenarios, we generated two variations of problem instances for each size, with different battery levels: 1%–5% (insufficient for small), 5%–10% (insufficient for medium and large) and 30%–100% (sufficient for all sizes). In cases of insufficient battery levels, recharging is necessary because AMRs have little remaining battery power at the beginning, whereas in problems with sufficient battery levels, recharging rarely occurs. In addition to

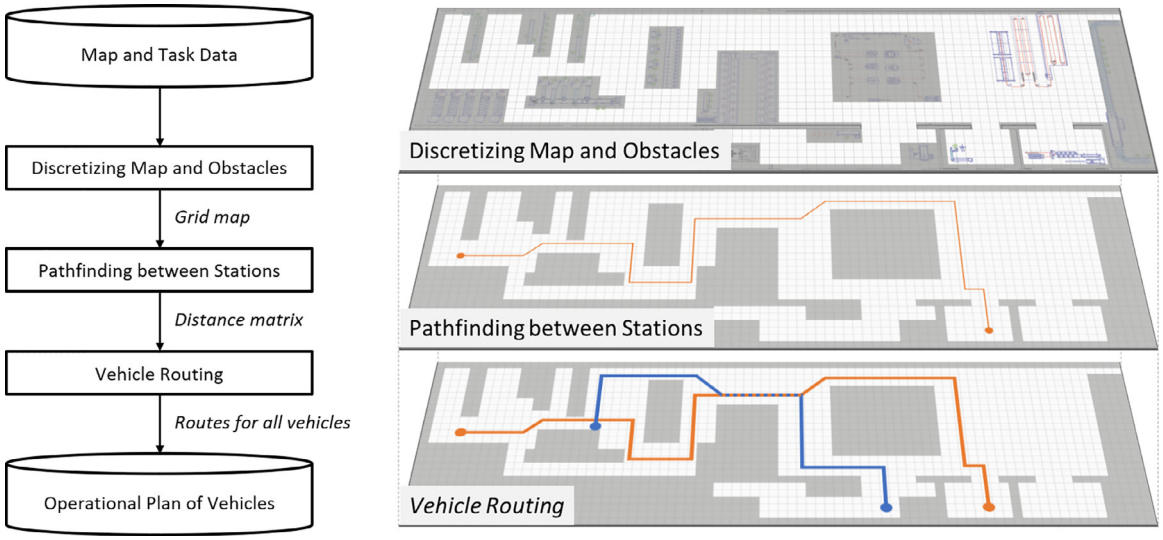
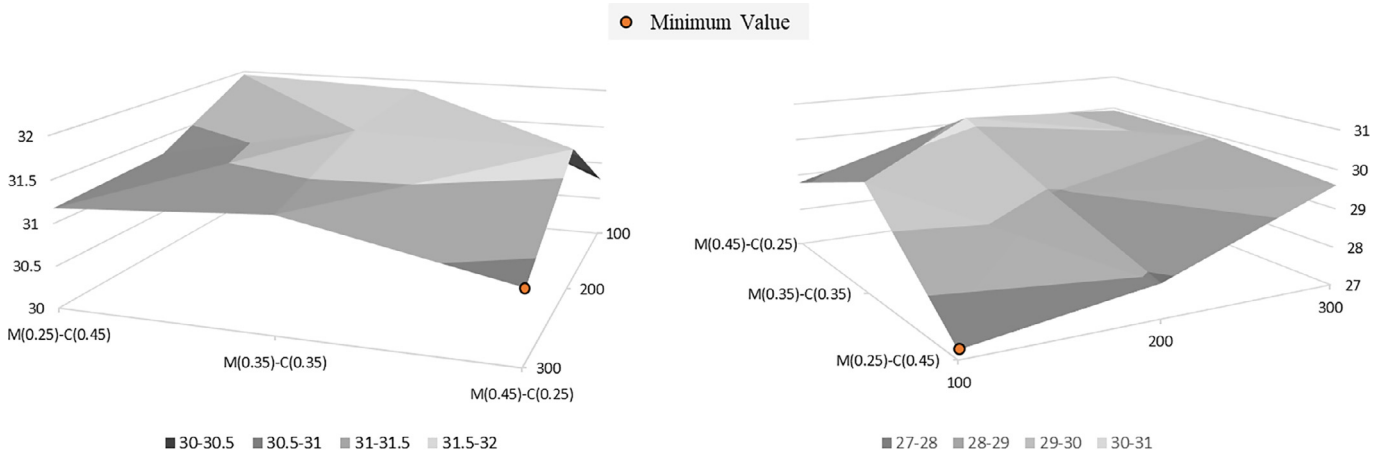


Fig. 8. Overview of the experimental design.



(a) Sensitivity of parameters for SGA

(b) Sensitivity of parameters for MA

Fig. 9. Sensitivity analysis of parameters for SGA and MA.

Table 5
Parameters for the algorithms.

Algorithm	Parameter	Value
SGA	Population size	300
	Crossover rate	0.25
	Mutation rate	0.45
	mv_num	3
	ms_num	2
	Number of survivors	90
	Tournament size	5
MA	Population size	100
	Crossover rate	0.25
	Mutation rate	0.45
	mv_num	3
	ms_num	2
	Tournament size	5
TRIGA	Range of *Threshold*	[0, 0.5]
	Threshold_Step	0.05
	Range of *Grouping_Size*	[1, 5]
VIGA	Range of *Grouping_Size*	[1, 5]

the average total tardiness, the relative deviation index (RDI) was used to compare the relative performances of the algorithms. The RDI of the k^{th} experiment was calculated using Eq. (26) (Akhshabi, Tavakkoli-Moghaddam, and Rahnamay-Roodposhti 2014):

$$RDI_k = \frac{F_k - Min_k}{Max_k - Min_k} \times 100 \tag{26}$$

where F_k represents the total tardiness obtained in the k^{th} experiment, and Max_k and Min_k represent the best and worst solutions in the kth experiment. The average total tardiness, computation times, and RDIs of all the algorithms for the 30 problem instances for each scenario are compared in Tables 6–8. Additionally, the average RDIs for the respective test sets and the best RDIs for each test set are presented in Fig. 10.

In the case of the mathematical model, two MILP models with partial and full recharging were denoted as MILP-P and MILP-F, respectively. The MILP models yielded the optimal solutions for the small problem (5 requests with 2 AMRs), but could not find the optimal solution for the medium-sized problem (10 requests

Table 6
Average performances of the algorithms for 30 small-sized problem instances (5 requests with 2 AMRs) with different battery levels.

	5 requests with 2 AMRs					
	Insufficient battery level			Sufficient battery level		
	$\sum T_i$	Computation time (s)	RDI (%)	$\sum T_i$	Computation time (s)	RDI (%)
MILP-P	2.48	2150.22	0	1.41	1385.67	0
MILP-F	10.63	2445.93	49.34	1.41	1120.57	0
SGA	13.16	300	64.64	1.41	300	0
NS	13.43	300	66.46	1.5	300	5.4
MA	13.17	300	64.69	1.42	300	0.55
TRIGA	15.87	<1	81.25	1.86	<1	38.75
VIGA	15.73	<1	80.68	1.86	<1	38.78
STD	18.69	<1	97.77	2.93	<1	95.1
EDT	16.46	<1	85.0	1.95	<1	45.24

Table 7
Average performances of the algorithms for 30 medium-sized problem instances (10 requests with 2 AMRs) with different battery levels.

	10 requests with 2 AMRs					
	Insufficient battery level			Sufficient battery level		
	$\sum T_i$	Computation time (s)	RDI (%)	$\sum T_i$	Computation time (s)	RDI (%)
MILP-P*	5.95	3600	0.03	5.35	3600	5.16
MILP-F*	10.81	3600	15.02	5.35	3600	5.16
SGA	10.98	300	15.72	5.27	300	3.73
NS	12.26	300	20.42	5.49	300	7.34
MA	10.73	300	15.04	5.18	300	2.06
TRIGA	26.67	<1	67.95	7.69	<1	45.76
VIGA	27.04	<1	69.51	7.77	<1	46.85
STD	36.54	<1	97.99	11.43	<1	100
EDT	27.71	<1	71.81	8.0	<1	50.55

* All solutions from MILP models were feasible solutions.

Table 8
Average performances of the algorithms for 30 large-sized problem instances (50 requests with 10 AMRs) with different battery levels.

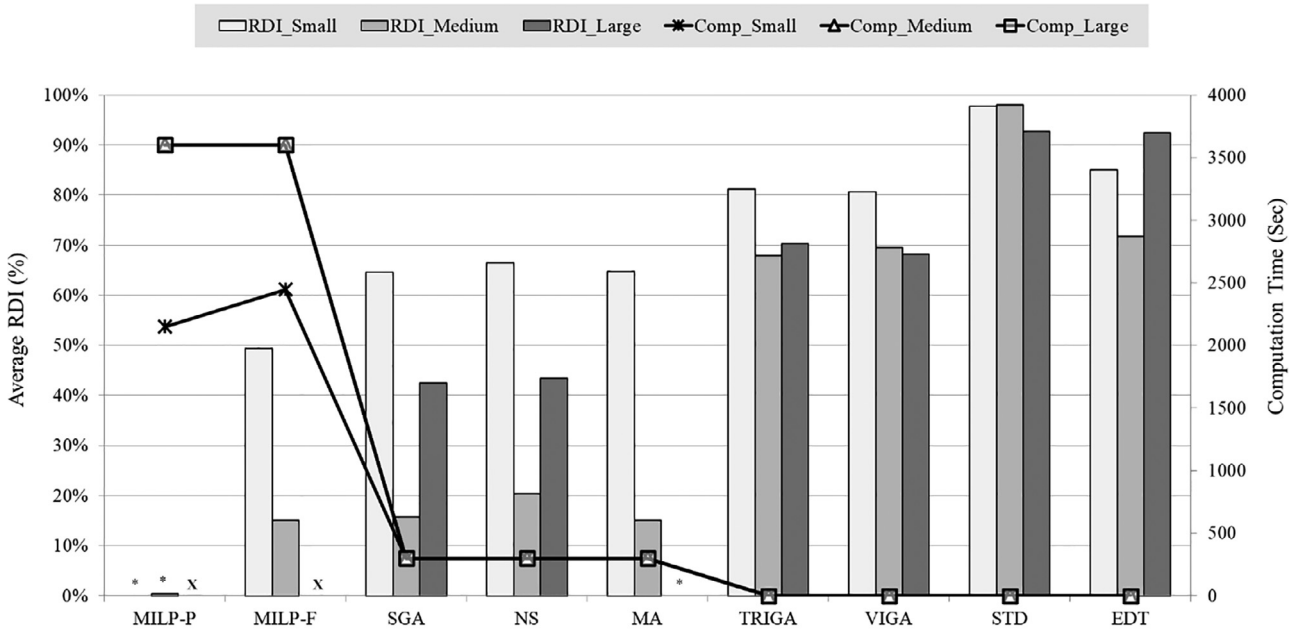
	50 Requests with 10 AMRs					
	Insufficient battery level			Sufficient battery level		
	$\sum T_i$	Computation time (s)	RDI (%)	$\sum T_i$	Computation time (s)	RDI (%)
MILP-P*	–	3600	–	–	3600	–
MILP-F*	–	3600	–	–	3600	–
SGA	92.14	300	42.55	31.59	300	22.06
NS	92.81	300	43.38	31.7	300	22.42
MA	47.81	300	0	25.87	300	0
TRIGA	120.59	<1	70.36	34.22	<1	32.21
VIGA	118.33	<1	68.15	33.45	<1	29.27
STD	146.29	<1	92.71	54.87	<1	99.12
EDT	143.37	<1	92.51	40.64	<1	57.84

* MILP models could not find any feasible solutions.

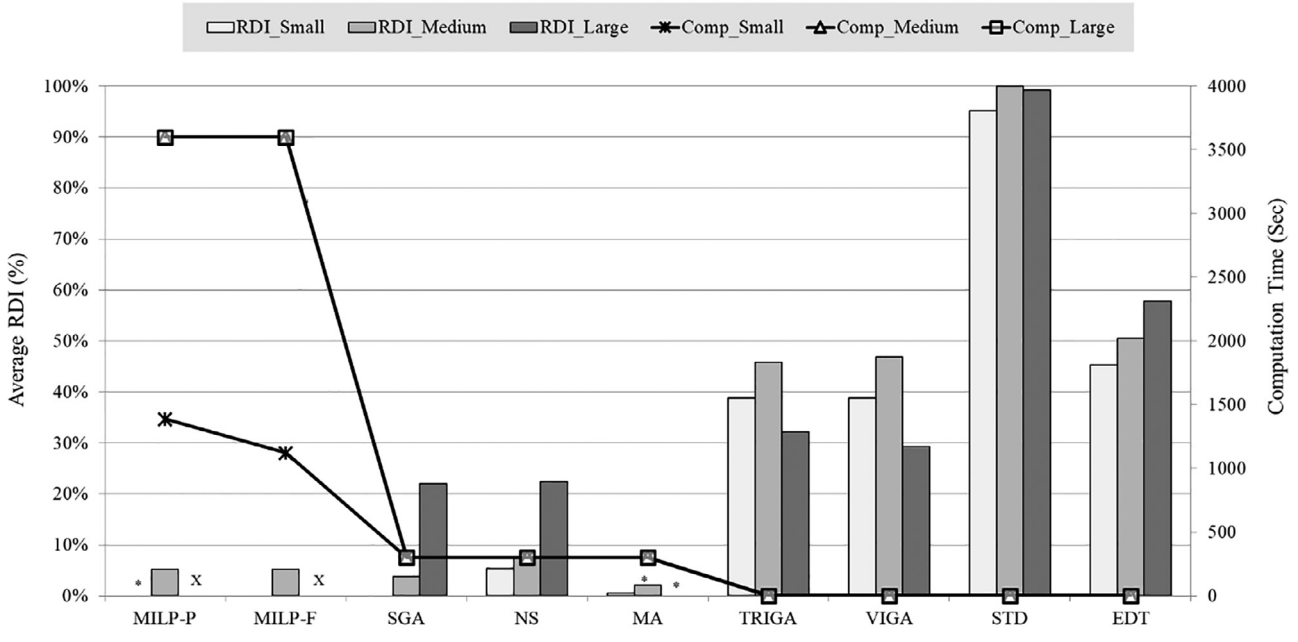
with 2 AMRs) under the same computation time limit. Especially, MILP-P outperformed MILP-F and the heuristic algorithms under insufficient battery levels, because it can determine the best level of battery to be recharged at a specific time, whereas the other algorithms are designed to fully charge the battery. Among the approaches with full recharging, the results indicated that the gap of the average RDIs between MILP-F and MA continued to decrease when the size of the problem increased. In the case of problem instances in which charging was not necessary (battery levels were sufficient), the gap between the MILP models and the best heuristic (MA) was insignificant (0.55 to 2.06%).

Although MILP-P yielded the best performances via partial recharging than MILP-F and the other heuristic algorithms, it encounters difficulties with complex problems in the real world. First, the MILP models merely found feasible solutions to the larger problems (50 requests with 10 AMRs) within an hour, even though it is capable of finding optimal solutions to small-sized problems. In addition, the high performance of MILP-P largely depends on the reduced charging time resulting from partial recharging. However, when an AMR visits a charging station, it is difficult to determine the exact amount of battery to be recharged for serving of upcoming transportation requests. Thus, well-designed heuristic and metaheuristic approaches are more suitable for solving complicated problems effectively.

Among the heuristic approaches, the results for the small-sized problems indicated that the SGA and MA could find near-optimal



a) Average RDIs and computation times under insufficient battery levels



b) Average RDIs and computation times under sufficient battery levels

Fig. 10. Average RDIs of algorithms for three test sets of problem instances with different battery levels. The best RDIs among the algorithms for each test set are denoted by an asterisk (*); a cross mark (X) represents the cases in which MILP could not produce any feasible solutions within the time limit.

solutions and that there was no significant difference between them, although the SGA produced the lowest average total tardiness. In the cases of the medium- and large-sized problems, the MA outperformed the other algorithms; moreover, the average total tardiness gap between the MA and the other algorithms continued to broaden as the problem size increased. Regarding the relative performance, the MA outperformed the other algorithms in that it yielded the smallest average RDI value for most cases, as shown in Fig. 10.

To verify the advantages of VIGA- and TRIGA-based initiation over fully random chromosomes, we compared the average performances of the largest problem set for two different initialisations, as shown in Fig. 11. In the case of MA, the results showed that the average performances were significantly improved, by between

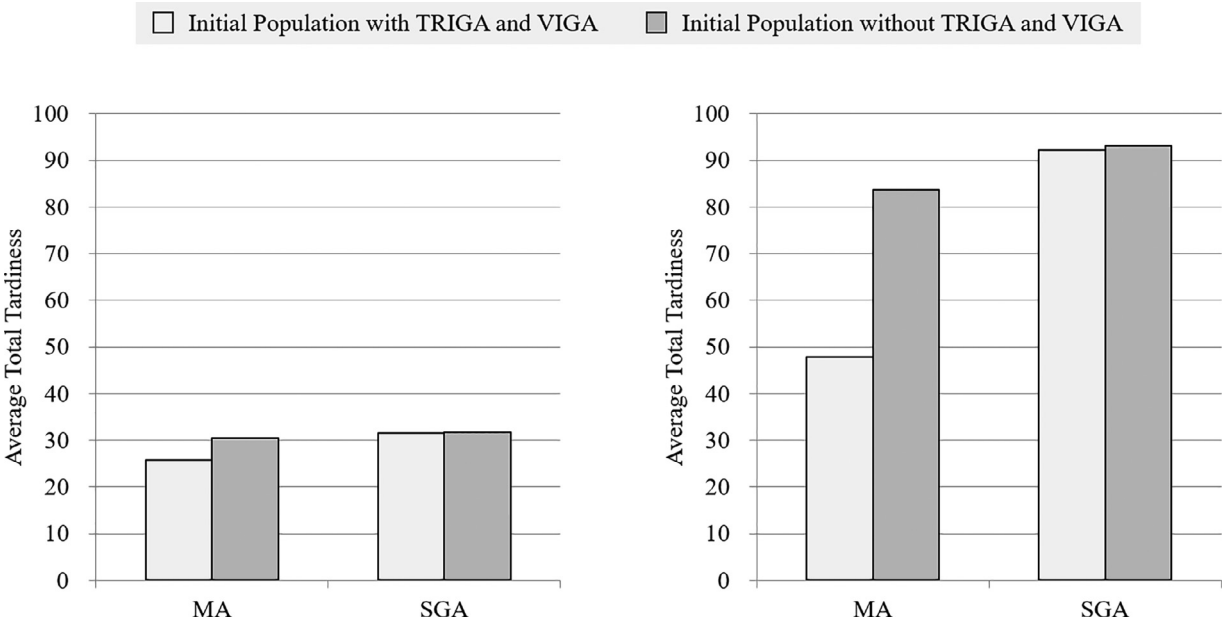


Fig. 11. Summary of average total tardiness of MA and SGA based on initial population with and without TRIGA and VIGA for large-sized problem instances (50 requests with 10 AMRs).

18 and 48%, by initialising the population with the proposed heuristics. The results also indicated that the combination of the initialisation and local search techniques of MA could produce better performances than SGA, especially under insufficient battery levels. Based on the results of the three test sets, we can conclude that the proposed VIGA and TRIGA can not only find good solutions quickly in reacting to changes, but also can significantly improve the performance of MA.

As an illustration of the routes of the AMRs with their performances, Fig. 12 represents the solutions of MILP-P, MILP-F, VIGA, EDT, GA, and MA for a small-sized problem (5 requests with 2 AMRs) along with the total recharging times, remaining energy capacities, and two different types of routes (brown routes for serving of requests and green routes for detours to visit recharging stations). The results showed that the proposed VIGA algorithm found a significantly better solution than did EDT, which is commonly used for minimising tardiness, and that MA effectively improved the performance of the populations initialised by VIGA and TRIGA. Among the algorithms, MILP-P produced the best solution, due to its partial recharging strategy, while MILP-F, GA, and MA could find the optimal solution with the full recharging strategy.

However, the MILP models with full and partial recharging strategies, notwithstanding its utility for saving recharging times and availability to guarantee the optimality, have some issues to be resolved before they can be considered to be applicable to real-world problems. First, the increased number of decisions that determines the time to visit a recharging station and the amount of battery level to be charged on each visit may complicate the vehicle routing problem, which is already NP-hard. As shown in Fig. 12, the computation times of the MILP models (2450 and 2337 s) were significantly greater than those of the other heuristics (less than 600 s), even for a small-sized problem instance.

Another issue is the small amount of battery capacity (close to the given limitation) after partial charging, especially for MILP-P. The average battery levels of MILP-P in Fig. 12 showed that partial recharging, at the expense of future recharging times, improves the performances by minimising the remaining battery level at the end of schedules. In the optimal solutions obtained by the MILP-P, the remaining battery levels of the vehicles were almost near the given l_k (5% of maximum capacity) for reduction of recharging times, and additional recharging was inevitable afterward. In the case of the heuristics, the remaining energy capacities after serving all of the requests were more adequate to serve upcoming requests.

In order to confirm the performance improvement by the recent advances in charging technologies as well as the benefits of partial recharging, we designed four experiments with different charging speeds for Fig. 12's illustrative example. In these experiments, we simulated different speeds of charging and battery swaps for AMRs based on specifications noted in the Appendix. As shown in Fig. 13, the results indicated that the performance gap of the average total tardiness between heuristics and MILP continued to narrow when the charging speed was increased, due to the reduced benefits from partial recharging. Thus, we concluded that the MILP with partial recharging is applicable only when the computation time is sufficiently long, the size of problems is small, all transportation requests are known, and no additional battery capacity above the safety level is required after serving all transportation requests. In addition, as the charging speeds of AMRs increase due to technology advances, the gap between MILP and our heuristics will be narrow further while the advantages of heuristics, such as ease of implementation and sufficient battery levels after recharging, will be kept. Thus, the proposed heuristics (TRIGA, VIGA, and MA) will prove to be more applicable in practice.

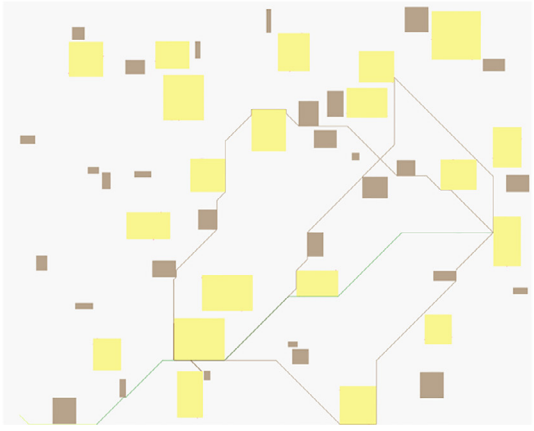
In summary, the simulation results indicated that the two heuristic algorithms—TRIGA and VIGA—offer higher performance than the other DRs and that they are suitable with regard to scalability for extremely large real-world problems and the ability to react to dynamic changes, owing to their high computation speed. Additionally, the proposed MA significantly improved the performance of the proposed heuristics by using more computation time when TRIGA and VIGA were applied for generation of an initial population. Finally, the simulation results demonstrated that the performance gap between MILP with partial recharging and heuristics with full recharging continued to tighten as the charging speed increased, due to technical advances such as quick chargers.

a) MILP-P (computation time = 2450 s)
$\sum T_i$ = 1.27, total recharging time = 0.32
Avg. remaining battery level = 5%

b) MILP-F (computation time = 2337 s)
$\sum T_i$ = 6.96, total recharging time = 2.88
Avg. remaining battery level = 46.79%

c) VIGA (computation time < 1s)
$\sum T_i$ = 7.0, total recharging time = 2.88
Avg. remaining battery level = 46.56%

d) EDT(computation time < 1s)
$\sum T_i$ = 16.66, total recharging time = 5.78
Avg. remaining battery level = 47.1%



e) GA and MA (computation time = 300 s)
$\sum T_i$ = 6.96, total recharging time = 2.88
Avg. remaining battery level = 46.79%

Fig. 12. Solutions and routes for small-sized problem with insufficient battery levels (5 requests with 2 AMRs). The yellow and brown rectangles indicate the work centres and obstacles, respectively. The green lines represent detouring routes for recharging, and the brown lines represent routes for serving of requests. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

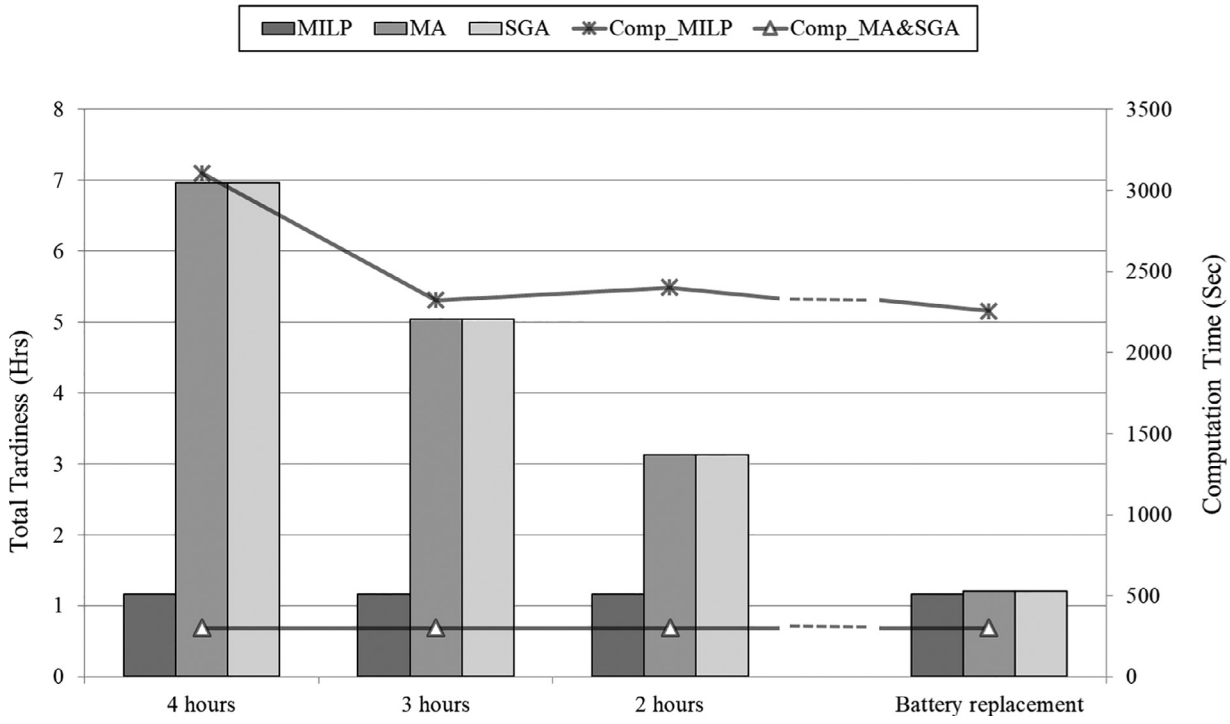


Fig. 13. Summary of performances with different charging technologies for Fig. 12's illustrative example.

6. Conclusion

We identified the characteristics and constraints of AMRs and formulated a new mathematical model with consideration of partial recharging and its detouring distances, for minimisation of the total tardiness of all transportation requests. To solve the defined problem practically, we proposed two constructive heuristic algorithms—the TRIGA and VIGA—that had high computation speed and were based on two different grouping strategies. We also proposed an MA for the exploration and exploitation of near-optimal solutions within a reasonable time. We evaluated the proposed algorithms in comparison with existing algorithms by conducting simulation experiments under various scenarios. The results indicated that the TRIGA and VIGA, compared with the other DRs, could find good solutions. Further, the MA outperformed NS and the SGA with regard to the average total tardiness and RDI.

The major contributions of this study were its addressing of the PDP with recharging and its development of (1) two constructive heuristics (TRIGA and VIGA) for finding good solutions quickly and (2) efficient evolutionary algorithm (MA) for improvement of the initial population using two constructive heuristics. Also, the proposed MILP with partial recharging can, by varying the charging rate, provide better understanding of the benefits of different charging technologies. Beyond traditional automation, smart factories, by introducing AMRs for material handling, connect all machines to make them more intelligent and flexible systems. Accordingly, the proposed algorithms can accelerate the adoption of such AMRs by improving the efficiency of smart factories. In addition to the area of material handling, the proposed mathematical model and algorithms have numerous smart-city applications, such as ride-sharing problems with shared autonomous electric vehicles.

Nevertheless, the proposed approach has some limitations. First, in this study, we addressed the problems of linear energy consumption and recharging; however, various factors such as weights and state of charge can significantly affect the rates of energy consumption and recharging. Additionally, we considered only static constraints on the shop floor, such as static obstacles and fixed due dates, though consideration of dynamic changes can be crucial in situations where a large number of uncertainties including breakdowns and moving objects exists (Shnits, Rubinovitz & Sinreich, 2004).

Future work can proceed in several directions. First, in order to deal with unexpected delays and satisfy the expected cycle time, the dynamic aspects of manufacturing environments, such as new obstacles, unexpected delays, and changes in due dates, are interesting and worthy of investigation, especially for verification of the proposed algorithm's performance under uncertainties. Additionally, relocation strategies for the minimisation of delays can be studied for cases in which vehicles are idle or are waiting for transportation requests. Furthermore, strategies for avoiding collisions while minimising the detouring distances and finding the best route with consideration of correlations with other paths can be investigated. Finally, various constraints including load-dependant or speed-dependant consumption and determination of charging stations' locations are interesting and worthy of investigation (He, Zhang & Nip, 2017; Lu, Zhou, Mahmoudi, Shi & Peng, 2019).

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ejor.2020.07.049.

References

Bechtsis, D., Tsolakis, N., Vlachos, D., & Srai, J. S. (2018). Intelligent autonomous vehicles in digital supply chains: A framework for integrating innovations towards sustainable value networks. *Journal of cleaner production, 181*, 60–71.

Berman, B. (2002). Should your firm adopt a mass customization strategy? *Business Horizons, 45*(4), 51–60.

Bruglieri, M., Mancini, S., Pezzella, F., Pisacane, O., & Suraci, S. (2017). A three-phase matheuristic for the time-effective electric vehicle routing problem with partial recharges. *Electronic Notes in Discrete Mathematics, 58*, 95–102.

Chand, P., Mishra, B. S. P., & Dehuri, S. (2010). A multi objective genetic algorithm for solving vehicle routing problem. *International Journal of Information Technology and Knowledge Management, 2*(2), 503–506.

Chen, T., Zhang, B., Pourbabak, H., Kavousi-Fard, A., & Su, W. (2016). Optimal routing and charging of an electric vehicle fleet for high-efficiency dynamic transit systems. *IEEE Transactions on Smart Grid, 9*(4), 3563–3572.

Cicirello, V. A. (2006, July). Non-wrapping order crossover: An order preserving crossover operator that respects absolute position. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation* (pp. 1125–1132). ACM.
Cordeau, J. F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research, 54*(3), 573–586.
Curtois, T., Landa-Silva, D., Qu, Y., & Laesanklang, W. (2018). Large neighbourhood search with adaptive guided ejection search for the pickup and delivery problem with time windows. *EURO Journal on Transportation and Logistics, 7*(2), 151–192.
Dumas, Y., Desrosiers, J., & Soumis, F. (1991). The pickup and delivery problem with time windows. *European Journal of Operational Research, 54*(1), 7–22.
Egbelu, P. J., & Tanchoco, J. M. (1984). Characterization of automatic guided vehicle dispatching rules. *International Journal of Production Research, 22*(3), 359–374.
Ganesh, K., & Narendran, T. T. (2007). CLOVES: A cluster-and-search heuristic to solve the vehicle routing problem with delivery and pick-up. *European Journal of Operational Research, 178*(3), 699–717.
He, Q., Zhang, X., & Nip, K. (2017). Speed optimization over a path with heterogeneous arc costs. *Transportation Research Part B: Methodological, 104*, 198–214.
Hiermann, G., Puchinger, J., Ropke, S., & Hartl, R. F. (2016). The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research, 252*(3), 995–1018.
Ho, Y. C., & Chien, S. H. (2006). A simulation study on the performance of task-determination rules and delivery-dispatching rules for multiple-load AGVs. *International journal of production research, 44*(20), 4193–4222.
Ho, Y. C., & Liu, H. C. (2009). The performance of load-selection rules and pick-up-dispatching rules for multiple-load AGVs. *Journal of Manufacturing Systems, 28*(1), 1–10.
Keskin, M., & Çatay, B. (2016). Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies, 65*, 111–127.
Keskin, M., & Çatay, B. (2018). A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. *Computers & Operations Research, 100*, 172–188.
Li, H., & Lim, A. (2003). A metaheuristic for the pickup and delivery problem with time windows. *International Journal on Artificial Intelligence Tools, 12*(02), 173–186.
Li, Y., Chen, H., & Prins, C. (2016). Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests. *European Journal of Operational Research, 252*(1), 27–38.
Lin, J., Zhou, W., & Wolfson, O. (2016). Electric vehicle routing problem. *Transportation Research Procedia, 100*(12), 508–521.
Lu, G., Zhou, X., Mahmoudi, M., Shi, T., & Peng, Q. (2019). Optimizing resource recharging location-routing plans: A resource-space-time network modeling framework for railway locomotive refueling applications. *Computers & Industrial Engineering, 127*, 1241–1258.
Macrina, G., Laporte, G., Guerriero, F., & Pugliese, L. D. P. (2019). An energy-efficient green-vehicle routing problem with mixed vehicle fleet, partial battery recharging and time windows. *European Journal of Operational Research, 276*(3), 971–982.
Mahmoudi, M., & Zhou, X. (2016). Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations. *Transportation Research Part B: Methodological, 89*, 19–42.
Masmoudi, M. A., Hosny, M., Demir, E., Genikomsakis, K. N., & Cheikhrouhou, N. (2018). The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation research part E: Logistics and transportation review, 118*, 392–420.
Material Handling Institute. (2018). *The 2018 MHI Annual Industry Report - Overcoming Barriers to NextGen Supply Chain Innovation*. Retrieved from https://www.mhi.org/publications/report.
Mitrović-Minić, S., Krishnamurti, R., & Laporte, G. (2004). Double-horizon based heuristics for the dynamic pickup and delivery problem with time windows. *Transportation Research Part B: Methodological, 38*(8), 669–685.
Muñoz-Carpintero, D., Sáez, D., Cortés, C. E., & Núñez, A. (2015). A methodology based on evolutionary algorithms to solve a dynamic pickup and delivery problem under a hybrid predictive control approach. *Transportation Science, 49*(2), 239–253.
Nanry, W. P., & Barnes, J. W. (2000). Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B: Methodological, 34*(2), 107–121.
Rodenberg, O. B. P. M., Verbree, E., & Zlatanova, S. (2016). Indoor A* pathfinding through an octree representation of a point cloud. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 4*, 249.
Savelsbergh, M. W., & Sol, M. (1995). The general pickup and delivery problem. *Transportation Science, 29*(1), 17–29.
Schiffer, M., & Walther, G. (2017). The electric location routing problem with time windows and partial recharging. *European Journal of Operational Research, 260*(3), 995–1013.
Schneider, M., Stenger, A., & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science, 48*(4), 500–520.
Shnits, B., Rubinovitz, J., & Sinreich, D. (2004). Multicriteria dynamic scheduling methodology for controlling a flexible manufacturing system. *International Journal of Production Research, 42*(17), 3457–3472.
Tan, K. C., Lee, L. H., Zhu, Q. L., & Ou, K. (2001). Heuristic methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering, 15*(3), 281–295.
Ting, C. K., Liao, X. L., Huang, Y. H., & Liaw, R. T. (2017). Multi-vehicle selective pickup and delivery using metaheuristic algorithms. *Information Sciences, 406*, 146–169.
Utamima, A., Pradina, K. R., Dini, N. S., & Studiawan, H. (2015). Distribution route optimization of gallon water using genetic algorithm and tabu search. *Procedia Computer Science, 72*, 503–510.
Wang, C., Wang, L., Qin, J., Wu, Z., Duan, L., Li, Z., et al. (2015, August). Path planning of automated guided vehicles based on improved a-star algorithm. In Proceedings of the IEEE International Conference on Information and Automation (pp. 2071–2076). IEEE.
Wang, H., & Cheu, R. L. (2013). Operations of a taxi fleet for advance reservations using electric vehicles and charging stations. *Transportation Research Record, 2352*(1), 1–10.
Wen, M., Linde, E., Ropke, S., Mirchandani, P., & Larsen, A. (2016). An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem. *Computers & Operations Research, 76*, 73–83.
Zhu, Z., Xiao, J., He, S., Ji, Z., & Sun, Y. (2016). A multi-objective memetic algorithm based on locality-sensitive hashing for one-to-many-to-one dynamic pick-up-and-delivery problem. *Information Sciences, 329*, 73–89.