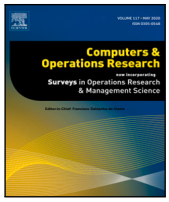
Computers & Operations Research 169 (2024) 106717

Contents lists available at ScienceDirect

Computers and Operations Research

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



An iterated local search matheuristic approach for the multi-vehicle inventory routing problem

Demetrio Laganà [a,*], Enrico Malaguti [b], Michele Monaci [b], Roberto Musmanno [a], Paolo Paronuzzi [b]

[a] DIMEG - Università della Calabria, Ponte Pietro Bucci, 87036, Cosenza, Italy
[b] DEI - Università di Bologna, Viale Risorgimento 2, 40136, Bologna, Italy

ARTICLE INFO

Keywords:
Multi-vehicle inventory routing
Matheuristic
Column generation
Local search

ABSTRACT

The multi-vehicle inventory routing problem considers an integrated system in which a supplier must satisfy deterministic demands from a set of customers over a finite and discrete time horizon. A limited inventory capacity is available at the customers, and a deterministic amount of product is available at the supplier in each period to fulfill customer demands with a homogeneous fleet of vehicles. The supplier decides when to resupply the customers, the quantities of product to deliver, and the routes to serve the customers. The aim is to find the best supply policy, which minimizes the total inventory and routing costs while ensuring that no stock-out occurs at the customers, while respecting the capacity of each vehicle. The problem has attracted significant attention in recent decades due to its wide applicability in fields where both inventory and routing aspects are addressed together.

In this work, we present a matheuristic algorithm based on a mathematical formulation in which we associate a decision variable with each route and period. The size of this set of decision variables is clearly exponential; therefore, we devise a column generation approach in which we heuristically generate a subset of these variables within an iterated local search framework. Each iteration of the algorithm is composed by two phases: in the first one, the current model is solved by means of a general-purpose solver, whereas in the second one the model is updated by replacing some variables with a suitable subset of new variables. Once a local optimum is reached, a diversification step takes place in which the set of columns is expanded to possibly define a new starting solution. We have compared our approach with state-of-the-art algorithms on a large benchmark of instances from the literature. Our computational results show that the proposed algorithm outperforms most of the existing heuristic methods.

1. Introduction

The management of a supply chain consists of controlling three main processes: the production of finite items from raw materials or basic components, inventory management of the products, and their distribution to customers. In the past, optimal decisions related to each phase were made separately. In the modern era, the impressive growth in computing power has led to the development of optimization algorithms focused on integrated problems, where strategic, tactical, and operational decisions related to two or more processes of supply chain management can be jointly considered. Solving integrated problems offers significant economic advantages for firms, as it happens for example when combining inventory and routing decisions in supply chains, which can result in substantial cost savings (Solyalı and Süral, 2017; Dušan et al., 2022). An integrated optimal solution, while more challenging to find, represents a global optimum compared to the suboptimal solutions that can be easily obtained by solving a sequence of independent optimization problems. Recent advances in scientific research focused on logistics have stimulated the study of even more integrated problems in supply chain management (Archetti and Speranza, 2016).

A business model that emphasizes integration and coordination in supply chain management is Vendor Managed Inventory (VMI), in which the supplier is responsible for monitoring and maintaining inventory levels at customers. This model can reduce inventory levels, minimizing associated costs, and prevent stock-outs in the supply chain. However, it requires a high level of coordination and collaboration among supply chain members. Within the VMI model, the optimization problem that integrates and coordinates tactical inventory management and operational vehicle routing decisions is known as the Inventory

* Corresponding author.
E-mail addresses: demetrio.lagana@unical.it (D. Laganà), enrico.malaguti@unibo.it (E. Malaguti), michele.monaci@unibo.it (M. Monaci), roberto.musmanno@unical.it (R. Musmanno), paolo.paronuzzi@unibo.it (P. Paronuzzi).

https://doi.org/10.1016/j.cor.2024.106717
Received 18 October 2023; Received in revised form 20 March 2024; Accepted 28 May 2024
Available online 31 May 2024
0305-0548/© 2024 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Routing Problem (IRP). The IRP is the challenge faced by a single vendor or supplier aiming to fulfill the demands of a set of customers over a finite and discrete time horizon by using a limited fleet of homogeneous vehicles (in the reminder of the paper, the terms *vendor* and *supplier* are used interchangeably). The supplier must ensure that no stock-outs occur at each customer while minimizing the total inventory (at the supplier's end and for the customers) and routing costs. In the deterministic setting, a given production rate is available at the supplier to meet customer demands. Both production and demands are known in advance for each period within the time horizon. From an operational perspective, each customer can be visited by one vehicle per period at most, and the combined demand of the customers served by each vehicle must not exceed its capacity.

The IRP has garnered significant attention from researchers in recent years, and has been associated with a wide variety of applications. Christiansen et al. (2011) studied an IRP within a maritime environment, where a heterogeneous fleet of bulk ships delivered multiple non-mixable cement products from suppliers to regional silo stations along the coast of Norway. Bertazzi et al. (2019) proposed a matheuristic for the multi-depot IRP applied to nanostores in a South East Asian city. These nanostores are part of the Procter & Gamble company supply chain, which is resupplied according to a VMI model. McKenna et al. (2020) investigated a VMI model used to resupply geographically dispersed units operating in a combat environment through unmanned aerial cargo vehicles. The impact of the resupply policies was evaluated in a hypothetical military scenario based on contingency operations in Afghanistan by the United States Army. The IRP is also addressed by Gaur and Fisher (2004) and Laganá et al. (2016) to find the optimal resupply policy for the distribution of products in supermarket chains. Popović et al. (2012) solved an IRP for fuel delivery in petrol station inventory management. Čankaya et al. (2019) defined an IRP for the distribution of humanitarian relief supplies in disaster operations management. Cárdenas-Barrón and Melo (2021) studied an IRP modeling the collection of waste vegetable oil in reverse logistics. Most applications focus on IRPs solved with deterministic optimization algorithms, which assume fully specified inputs.

This manuscript addresses the IRP, which involves deterministic demands that arise over a planning horizon and can be served by multiple vehicles, referred to as the **MIRP** hereafter. For this problem, our contributions to the literature include:

- A route-based mathematical formulation for the **MIRP** where variables associated with routes model the scheduling of the visits at the customers. The advantage of our formulation relies on the possibility to build routes fulfilling the routing requirements and to adapt them to the quantities shipped to the visited customers according to the inventory constraints. This flexibility allows to design a two-phase matheuristic that is very effective to solve medium and large **MIRP** instances from the literature. We refer to our matheuristic as ILS-M in the rest of the paper.
- Extensive computational results on benchmark instances and a comparison of ILS-M against state-of-the-art methods from the literature. The results prove that our algorithm outperforms the other heuristic methods from the literature on solution quality and computing time, and this is the case for both of the two considered resupply policies (Maximum-Level and Order-Up-to-Level policy).

The rest of the paper is organized as follows. Section 2 describes the problem. Section 3 provides a comprehensive literature review of the most recent work on the IRP. Section 4 presents a novel mathematical formulation for the **MIRP**. The ILS-M is described in Section 5. Section 6 presents the results of the computational study. Finally, Section 7 closes the paper with conclusions and discusses future work.

2. Problem definition

The **MIRP** is defined over a complete undirected graph $G = (V = N \cup \{0\}, E)$, where vertex 0 is the supplier and N is the set of customers with $|N| = n$. The deliveries to the customers are performed over a finite and discrete time horizon $T = \{1, \dots, H\}$ by means of a homogeneous fleet of m vehicles, each having a capacity Q. For each period $t \in T$, a known quantity p_t is available at the supplier to replenish customers in the period, so as to avoid stock-out at the customers and at the supplier itself. Each customer $i \in N$ has a maximum inventory capacity U_i and is associated with a non-negative demand d_{it} for each period $t \in T$. In addition, each node $i \in V$ has an initial inventory level I_{i0} and an inventory holding cost h_i for each unit of inventory carried at the end of a period. Finally, each edge $(i, j) \in E$ is associated with a transportation cost $c_{ij} \geq 0$, which contributes to the objective function every time a vehicle traverses that arc.

Multiple trips and split deliveries in each period of the planning horizon are not allowed, i.e., in each period, each vehicle may perform at most a trip and each customer may be visited by at most one vehicle.

The problem asks to determine a delivery plan that minimizes the sum of routing cost and inventory costs at the customers and supplier over the planning horizon, while avoiding stock-out at the customers and supplier.

In the most general version of the problem, the quantity shipped to a customer in each period is arbitrary, provided that the maximum inventory level is not exceeded, which also considers the inventory level at the end of the previous period. This replenishment policy corresponds to the so-called Maximum-Level (ML) policy. Unless specified, we will always refer to this version of the problem throughout the paper. However, we mention that an alternative replenishment policy, referred to as the Order-Up-to-Level (OU) policy, has been widely considered in the literature. In the OU policy, the quantity shipped to each customer in each period is constrained to be such that the maximum inventory level of the customer is reached, taking into account the inventory level at the end of the previous period.

3. Literature review

We start the literature review on the IRPs from the pioneering paper of Bell et al. (1983), in which IRP is introduced and motivated by an application concerning the inventory management of industrial gases at the customer locations combined with vehicle scheduling and dispatching. In this example they do not account for the inventory costs. After this paper, a very large number of contributions focused on the IRPs appeared in the literature. To date, surveys on the IRPs are presented by Campbell et al. (1998), Moin and Salhi (2007), Bertazzi et al. (2008), Andersson et al. (2010) and Coelho et al. (2014). Bertazzi and Speranza (2012, 2013) provide a tutorial on the IRPs. We refer the reader to these references for an in-depth overview of the main contributions on the IRP and its variants. New challenges in the field of IRPs can be found in the paper of Archetti and Bertazzi (2021). In the next sections, we provide an overview of the most significant contributions to inventory routing problems, categorized into exact and heuristic methods.

3.1. Exact methods

Our review of the exact approaches builds on the paper by Archetti et al. (2007), in which a branch-and-cut algorithm for the single-vehicle IRP is introduced. This approach considers both ML and OU policies, and is based on several classes of valid inequalities. Solyalı and Süral (2011) improve this formulation by reformulating the retailers' inventory replenishment decisions as a shortest-path problem solvable in $\mathcal{O}\left(H^2\right)$ time. Coelho et al. (2012) first formulate the **MIRP** as Mixed Integer Linear Program (MILP). Adulyasak et al. (2014) study different MILP formulations for the multi-vehicle Production Routing Problem

(PRP) of which the **MIRP** is a special case. They propose formulations with and without vehicle index, adapt some known valid inequalities to these formulations, and solve them with a tailored branch-and-cut algorithm by using an initial solution provided by a heuristic. Coelho and Laporte (2013) solve several classes of the **MIRP** by formulating them as a MILP. They design an improved branch-and-cut algorithm whose performance is evaluated on multi-vehicle instances derived from those proposed by Archetti et al. (2007, 2012) for the single-vehicle case. Archetti et al. (2014) provide a computational study of the **MIRP** formulations by extending the single-vehicle formulation proposed by Archetti et al. (2007) and Solyalı and Süral (2011) for the IRP. Desaulniers et al. (2016) propose a Dantzig–Wolfe decomposition approach to the **MIRP** and develop a branch-price-and-cut algorithm. Manousakis et al. (2021) present a novel two-commodity flow formulation for the **MIRP**. They define a new set of valid inequalities and develop a branch-and-cut algorithm. Guimarães et al. (2023) propose a vehicle-indexed formulation for the **MIRP** and a branch-and-cut algorithm in which a mechanism for recovering feasibility and improving partial solutions is embedded. Recently, Schenekemberg et al. (2023) present a parallel branch-and-cut algorithm for the PRP and the **MIRP**. Finally, we mention the recent work of Skålnes et al. (2023), where an effective solution method for the **MIRP** is presented; this approach consists in a branch-and-cut embedding a matheuristic algorithm that is executed every time a new primal solution is found.

3.2. Heuristic methods

In this section we review in detail the most recent and relevant contributions focusing on heuristic methods for the **MIRP**. We remark that many heuristics for the **MIRP** are based on metaheuristic and matheuristic schemes. Moin et al. (2011) investigate a many-to-one IRP in which the demand for each product is deterministic and time varying. They propose a mathematical formulation of the problem providing lower and upper bounds, and a hybrid genetic algorithm for solving large scale instances. Coelho et al. (2012) study a **MIRP** with consistency requirements on quantities, vehicles filling rate, and drivers. They develop an adaptive large neighborhood search algorithm with exact subproblems. Archetti et al. (2012) propose a tabu search based matheuristic for the **MIRP**. They solve single vehicle IRP instances with OU and ML policies. Santos et al. (2016) study the **MIRP** and propose a decomposition approach in which the routing subproblem is solved by using an iterated local search metaheuristic. The inventory subproblem is solved with an exact procedure based on mathematical programming. Archetti et al. (2017) design a matheuristic using a tabu search in combination with mathematical formulations to solve the **MIRP**. Alvarez et al. (2018) study a **MIRP** in which two objectives are addressed: the sum of the total inventory and routing cost, and the logistic ratio defined as the total travel cost divided by the total quantity delivered to customers. They develop metaheuristic algorithms based on iterated local search and simulated annealing. Chitsaz et al. (2019) study the assembly routing problem (ARP), and devise a three-phase matheuristic that decomposes the ARP into subproblems that can be solved exactly with mathematical formulations. The proposed approach is tested on production and inventory routing problem instances. Diniz et al. (2020) consider the **MIRP**, for which they investigate the impact of efficiently solving a recurring network flow problem related to the optimal replenishment of inventory at each customer in each period whenever delivery paths are defined. Vincent et al. (2021) propose a simulated annealing based algorithm to solve the multi-vehicle cyclic IRP. Archetti et al. (2021) present a matheuristic algorithm for the **MIRP** that uses the information obtained by a tabu search procedure to build MILP problems of small size that are solved sequentially. Vadseth et al. (2021) propose a matheuristic that, after the generation of a giant tour, adopts local search operators to create routes that are used in a path-flow formulation. The matheuristic iterates between solving the path-flow model with a small set of routes and updating the routes on the basis of the optimal solution generated in the previous iteration. We observe that the approach presented in our paper has several similarities to this algorithm. However, the diversification technique designed in our work, based on mathematical programming, is very different and more effective than that used in this matheuristic. Solyalı and Süral (2022) propose a matheuristic for the **MIRP** which is based on a sequential solution of three different MILPs modeling a restricted version of the problem, followed by the solution of Capacitated Vehicle Routing Problems (CVRPs) or Traveling Salesman Problems (TSPs).

4. A mathematical formulation: rq-MIRP

In this section, we introduce a mathematical formulation for the **MIRP**. The main feature of this formulation is that routing decisions are decoupled by packing and inventory decisions.

More into details, routing decisions are associated with variables describing the set of customers visited by a vehicle. With an abuse of notation, we will identify each subset of customers with a *route*, i.e., with a collection of edges visiting all customers in the subset with minimum routing cost.

In the following, we will denote by $\mathcal{R}$ the family of all possible routes (one for each subset of customers) and by c_r the routing cost for each route $r \in \mathcal{R}$.

Then, the formulation introduces three sets of variables:

- I_{it} denotes the inventory level at the end of period t at customer $i \in V$;
- q_{it} represents the quantity of product delivered to customer $i \in N$ in period $t \in T$;
- y_{rt} is a binary variable equal to 1 if route $r \in \mathcal{R}$ is used in period $t \in T$, and 0 otherwise.

A valid formulation for **MIRP** reads as follows:

$$\textbf{(rq-MIRP)}\ \min \quad \sum_{t\in T}\sum_{i\in V} h_i I_{it} + \sum_{t\in T}\sum_{r\in\mathcal{R}} c_r y_{rt} \tag{1}$$

$$\text{s.t.} \quad I_{0,t-1} + p_t - \sum_{i\in N} q_{it} = I_{0t} \qquad t \in T, \tag{2}$$

$$I_{i,t-1} - d_{it} + q_{it} = I_{it} \qquad i \in N,\ t \in T, \tag{3}$$

$$\sum_{r\in\mathcal{R}} y_{rt} \le m \qquad t \in T, \tag{4}$$

$$\sum_{r\in\mathcal{R}:\, i\in N(r)} y_{rt} \le 1 \qquad i \in N,\ t \in T, \tag{5}$$

$$Q(\{i\}) \sum_{r\in\mathcal{R}:\, i\in N(r)} y_{rt} \ge q_{it} \qquad i \in N,\ t \in T, \tag{6}$$

$$Q(N(r)) \sum_{r'\in\mathcal{R}(r)} y_{r't} \ge \sum_{i\in N(r)} q_{it} \qquad r \in \mathcal{R},\ t \in T, \tag{7}$$

$$I_{0t} \ge p_t \qquad t \in T, \tag{8}$$

$$0 \le I_{it} \le U_i - d_{it} \qquad i \in N,\ t \in T, \tag{9}$$

$$q_{it} \ge 0 \qquad i \in N,\ t \in T, \tag{10}$$

$$y_{rt} \in \{0,1\} \qquad r \in \mathcal{R},\ t \in T, \tag{11}$$

where, for any $S \subset N$, $Q(S) = \min\left\{Q, \sum_{i\in S} U_i\right\}$ denotes the maximum amount of product that can be delivered to the customers in S. In addition, for a given route r, we denote by $N(r)$ the set of customers belonging to route r and by $\mathcal{R}(r)$ the set of routes having at least one customer in common with r (including r itself).

The objective function minimizes the sum of the inventory costs and the costs of the selected routes. Eqs. (2) and (3) impose the inventory balance for both the depot and the customers, respectively. Constraints (4) limit the number of routes per period to the number of available vehicles, and (5) impose each customer be visited at most once per period. Inequalities (6) impose that a customer receiving a non-zero amount of product in a period must be visited by at least one route in that period. Each one of inequalities (7) imposes the capacity constraint for a route r at period t, when the route is used, i.e., $y_{rt} = 1$. In this case, due to (5), $y_{r't} = 0$ for all $r' \in \mathcal{R}(r) \setminus r$. When, instead, $y_{rt} = 0$, the

constraint allows customers belonging to r to be served by other routes in $\mathcal{R}(r)$. Finally, constraints (8) and (9) impose bounds on the inventory levels, and constraints (10) and (11) define the domain of the q and y variables.

In case of OU policy, the following constraints must be added to the formulation:

$$q_{it} \geq U_i \sum_{r \in \mathcal{R}: i \in N(r)} y_{rt} - I_{i,t-1} \quad i \in N,\ t \in T, \tag{12}$$

imposing that the quantity of product shipped to customer $i \in N$ in period $t \in T$ is $U_i - I_{i,t-1}$ if i is served by some route in period t.

Finally, we observe that the path-flow formulation introduced by Tinič et al. (2021) for the IRP with time windows and later extended by Vadseth et al. (2021) and Skålnes et al. (2023) to **MIRP** makes use as well of variables associated with routes. However, the variables in our model are a proper subset of those used in the aforementioned formulations, which require an additional set of flow variables associated with each arc and period to account for vehicle capacity constraints. A different formulation in which variables are associated with routes has been proposed in Desaulniers et al. (2016), where the authors introduce the concept of route delivery pattern to represent the quantity delivered to each customer along that route. The major difference between the formulation in Desaulniers et al. (2016) and **rq-MIRP** is that the former introduces one variable for each route, period, and route delivery pattern, whereas in our model decisions regarding the quantity to be delivered to each customer are separated from the route variables.

4.1. A simpler problem

In this section, we introduce a related yet simpler auxiliary problem where we relax the routing constraints of the problem and focus on calculating solutions in which multiple visits to a customer within a given period are allowed.

In our solution approach, the resulting problem, which is not computationally demanding in practice, is solved several times, using two different objective functions: (i) the first one minimizes the total number of visits to the customers, while (ii) the second one minimizes the sum of the inventory costs and of a proxy of the routing costs. The auxiliary problem is intended to define compact clusters that fulfill inventory requirements while grouping customers based either (i) on the number of visits required, or (ii) on both inventory and approximate routing costs.

In order to model the problem, besides variables q_{it} and I_{it}, we introduce binary variables z_{it} assuming value 1 if customer i receives some amount of product in period t, and 0 otherwise. The mathematical formulation of this problem is as follows:

$$\textbf{(AUX)} \min \quad \sum_{t \in T} \sum_{i \in V} z_{it} \tag{13}$$

$$\text{s.t.} \quad (2) - (3)$$

$$\sum_{i \in N} q_{it} \leq mQ \qquad t \in T \tag{14}$$

$$q_{it} \leq \min\{Q, U_i\} z_{it} \qquad i \in N, t \in T \tag{15}$$

$$(8) - (10)$$

$$z_{it} \in \{0,1\} \qquad i \in N,\ t \in T, \tag{16}$$

where objective function (i) is described by (13), in which the total number of visits at the customers over all periods is minimized. As to the second objective function (ii), it reads as

$$\min \quad \sum_{t \in T} \sum_{i \in V} h_i I_{it} + \sum_{t \in T} \sum_{i \in V} 2\, c_{0i}\, z_{it} \tag{17}$$

and sums the total inventory costs and the routing costs that would be incurred if customers are always served alone in a route starting and ending at the supplier.

In addition to inventory constraints (2) and (3), constraints (14) impose that the overall quantity delivered to the customers is not larger than the aggregate fleet capacity. In other words, the model produces a delivery schedule which may require a customer to be visited by more than one vehicle in a day. Constraints (15) limit the amount of product delivered to each customer in each period, possibly activating the corresponding z_{it} variable. Finally, constraints (16) define the domain of the variables.

In case the OU policy is considered, the following constraints must be added to the formulation:

$$q_{it} \geq U_i z_{it} - I_{i,t-1} \quad i \in N,\ t \in T, \tag{18}$$

imposing the quantity of product shipped to customer $i \in N$ in period $t \in T$ be $U_i - I_{i,t-1}$ if i is served in period t, and zero otherwise.

Finally, as already mentioned, our solution approach solves model **AUX** many times. Thus, the following no-good cut

$$\sum_{t \in T} \sum_{i \in N: \bar{z}_{it}=1} (z_{it} - 1) + 1 \leq 0$$

may be added to the formulation to prevent a previously obtained solution to be considered again.

5. A matheuristic for the rq-MIRP formulation

Formulation **rq-MIRP** is an exact formulation of the **MIRP** provided one enumerates all its variables and constraints. However, the size of the model is not polynomial with respect to the input given the exponential number of variables y_{rt} and constraints (7). Thus, defining the complete model and directly solving it turns out to be possible for very small instances only, even when using a column generation framework. Therefore, we develop an iterative heuristic algorithm that, at each iteration, operates with a limited subset of y_{rt} variables, and with the corresponding constraints (7). This subset includes all variables that define the best solution at the previous iteration, and additional variables aimed at enhancing the incumbent solution value. The resulting model implicitly defines a *neighborhood* of the current solution. This neighborhood includes all feasible solutions that can be obtained by combining the variables in the current best solution with the remaining ones, and can be efficiently explored provided the number of additional variables is limited. More into detail, the algorithm starts with an initialization phase in which an initial pool of variables are defined. Then, the algorithm enters a local search loop where (i) model **rq-MIRP** is defined with the current set of variables and solved it by using a MILP solver, and (ii) the model is updated by removing variables associated with routes that do not belong to the current incumbent solution and by generating new variables based on routes in the current solution. This loop is repeated until no solution improvement is experienced. When further improvement is not possible, a diversification step is executed in which new variables are generated so as to escape local minimum. Then, the local search loop is restarted.

This approach results in a matheuristic algorithm, akin to an iterated local search, referred to as Iterated Local Search Matheuristic or simply ILS-M hereafter.

Algorithm 1 Iterated Local Search Matheuristic.

```
Initialization: generate an initial pool of variables;
while stopping criteria do
    repeat
        solve model rq-MIRP;
        update the pool of y_rt variables;
    until solution improved
    diversify the pool of y_rt variables;
end while
```

5.1. Initialization

We initialize our algorithm by solving model **AUX**. Given an optimal solution $(\bar{\mathbf{q}}, \bar{\mathbf{I}})$ of the model, one can derive, for each period $t \in T$, a set

of quantities $\bar{q}_{it}$ to be delivered as well as a set of inventory levels $\bar{I}_{it}$. The former quantities induce, for each period t, an instance of CVRP which can be solved by means of any (exact or heuristic) algorithm for that problem. A feasible solution for **MIRP** can immediately be derived in case each CVRP instance admits a feasible solution with at most m vehicles.

Otherwise, we consider each infeasible CVRP instance and relax the constraint on the maximum number of available vehicles. In this case, though no feasible solution to **MIRP** is immediately available, a set of routes is produced, where each route r can be used to define a variable y_{rt} for each period $t \in T$, allowing to initialize model **rq-MIRP** accordingly. If the resulting **rq-MIRP** model turns out to be infeasible, we repeatedly solve **AUX**, alternating between the two objective functions discussed in Section 4.1 and introducing no-good cuts until **rq-MIRP** becomes feasible.

5.2. Local search

Once the algorithm is initialized, we iteratively repeat a phase in which model **rq-MIRP** is solved and the pool of variables is updated (steps 3–6). This phase works as a local search in which new columns y_{rt} are generated at each iteration. Given a feasible solution $(\bar{\mathbf{y}}, \bar{q})$, we remove from the model all the variables y_{rt} such that $\bar{y}_{rt} = 0$. Then, we use three different strategies to generate new columns.

In the first strategy, we solve a CVRP instance for each period $t \in T$ using quantities $\bar{q}_{it}$ as the demands. The routes of each solution are used to generate new variables y_{rt}. Similarly to the initialization, we first solve the CVRP by imposing that at most m vehicles be used, and remove this limitation if no CVRP feasible solution is found.

In the second strategy, for each variable $\bar{y}_{rt}$ of the incumbent solution, we generate a new variable y_{rp} for each period $p \in T$, $p \neq t$, corresponding to the same route r.

Lastly, the third strategy explores potential customer reallocations among routes within the current solution. For each period t and for each route r such that $\bar{y}_{rt} = 1$, and for each customer i of the route, we generate one variable $y_{r't'}$ in the same period for a route r' associated with customers $N(r') = N(r) \setminus \{i\}$. Then, for all the remaining periods $t' \in T, t' \neq t$ where customer i is not served, we generate one variable $y_{r't'}$ associated with route r' obtained by including customer i into one route, say $\rho \in \mathcal{R}$, such that $\bar{y}_{\rho t'} = 1$ and $\sum_{j \in N(\rho)} \bar{q}_{jt'} + \bar{q}_{it} \leq Q$, i.e., one route with sufficient residual capacity to accommodate the quantity $\bar{q}_{it}$. If more than one such route exists, we select the one with the lowest insertion cost. If no such route exists, we generate the variable associated with the route visiting customer i alone in period t'.

Algorithm 2 reports the pseudo-code of the customer reallocation amonrg routes. Function $g(r, i)$ returns the cheapest insertion cost of customer i into the existing route r.

Algorithm 2 Customer reallocation.

```
1: for t ∈ T do
2:     for r ∈ R : ȳ_rt = 1 do
3:         for i ∈ N(r) do
4:             generate y_r't with N(r') = N(r) \ i;
5:             for t' ∈ T : t' ≠ t and q̄_it' = 0 do
6:                 set Ω = {ρ ∈ R : ȳ_ρt' = 1 and Σ_{j∈N(ρ)} q̄_jt' + q̄_it ≤ Q}
7:                 if Ω ≠ ∅ then
8:                     r'' = argmin {g(ρ, i) : ρ ∈ Ω};
9:                     generate y_r't' with N(r') = N(r'') ∪ {i};
10:                else
11:                    generate y_r't' with N(r') = {i};
12:                end if
13:            end for
14:        end for
15:    end for
16: end for
```

At each iteration, we generate columns by means of all the three column generation strategies mentioned above. When the resulting model fails to produce an improved solution, we resort to an extended version of strategy 3 which allows for the re-allocations of sub-routes, i.e., portions of a route that include at least two subsequent customers. In other words, we focus on inserting a sequence of customers, rather than a single customer, into a different route. To this aim, we evaluate the insertion cost of the sequence of customers into each existing route, without changing the order of the customers in the sequence nor in the route, and select the route with the smallest insertion cost. Initially, we consider sequences of two customers and then increase their size until we either find an improved solution or reach a maximum size. Preliminary experiments have indicated that considering sequences with more than three customers is rarely beneficial. Whenever we discover a new solution, we add a no-good cut to formulation **rq-MIRP** as follows:

$$\sum_{t \in T} \sum_{r \in \mathcal{R} : \bar{y}_{rt} = 1} (y_{rt} - 1) + 1 \leq 0,$$

thus preventing the same solution from being used in future iterations.

5.3. Diversification

We continue the local search loop as long as we can improve the solution of **rq-MIRP**. When further improvement is not possible, we enter a diversification phase. In this phase, we enrich the pool of variables by solving **AUX** using the two objective functions described in Section 4.1. We solve the CVRP instances associated with the solutions and generate the corresponding y_{rt} variables, as we did in the initialization phase.

6. Computational experiments

In this section, we compare the performance of our solution algorithm with that of other methods solving the same problem. As previously mentioned in the introduction, many papers have addressed this problem in recent years, proposing both exact and heuristic approaches. Covering all these algorithms would likely result in an overly dense analysis and, in any case, would be beyond the scope of this paper. We assess the performance of our matheuristic algorithm by using the set of instances proposed by Archetti et al. (2007, 2012) for the single-vehicle IRP and later extended to **MIRP** by Coelho et al. (2012). All state-of-the-art methods are evaluated on this benchmark, which is typically divided into two classes (*small* and *large* instances). Therefore, for our computational analysis, we have chosen to include only those methods that can be considered Pareto optimal in terms of both average solution quality and average computing time for at least one of the two classes of instances. These methods are:

- the simulated annealing algorithm proposed by Alvarez et al. (2018), referred as AL-SA in the sequel;
- the matheuristic algorithm proposed by Diniz et al. (2020), for small instances only, referred as D-M in the sequel;
- the matheuristic algorithm proposed by Vadseth et al. (2021), referred as V-M in the sequel;
- the branch-and-cut proposed by Skålnes et al. (2023), referred as SV-BCM in the sequel.

We note that the first three methods are all heuristic algorithms, whereas the last one (SV-BCM) is the only exact method and requires a much larger computational effort than that of the heuristic approaches. Indeed, this algorithm has been included in the analysis for the sake of completeness and only to provide a reference.

6.1. Description of the instance benchmark

The depot and the customers correspond to arbitrary points on a 500 × 500 grid and the transportation cost between two points is an

integer rounding of their euclidean distance. The benchmark includes instances with two types of inventory costs, denoted as *low* and *high*. In the former case, inventory costs are randomly generated real numbers in the interval $[0.01, 0.05]$, whereas in the latter, inventory costs are randomly generated in the interval $[0.10, 0.50]$. The considered values for the number of vehicles are $\{2, 3, 4, 5\}$. The time-horizon consists of three or six periods, for small instances, while it always has six periods in large instances. Finally, small instances have a number of vertices ranging from 5 to 50 for $H = 3$, and from 5 to 30 for $H = 6$. In both cases, the number of vertices increases by 5 units from one group of instances to the next one. For large instances, we have $|V| \in \{50, 100, 200\}$. For each combination of the mentioned dimensions, there are five small instances and 10 large instances. This results in a benchmark of 640 small instances (among which two are excluded from the analysis, since they turn out to be infeasible) and 240 large instances. All benchmark instances we used, as well as the obtained solutions, are available at https://github.com/paoloparonuzzi/MIRP.git.

6.2. Computational setting

Our algorithm is implemented in C++ and use the CPLEX 22.10 MILP solver to tackle models **rq-MIRP** and **AUX**. Problems are solved in single-threaded mode. We set to 1 (emphasize feasibility over optimality) the MIP emphasis switch, while all the remaining CPLEX parameters are left to their default values. The experiments are performed on a computer equipped with an AMD Ryzen 3960X processor clocked at 3.8 GHz, 128 GB RAM, running Linux.

When solving a CVRP instance, we use the hybrid genetic algorithm (HGS-CVRP) of Vidal (2022), stopped after 500 iterations without improvement in the best solution. In addition, whenever a new y_{rt} variable is added to model **rq-MIRP**, we use Concorde (Applegate et al., 2007) to define the optimal order for visiting its customers and to compute the cost of the corresponding route.

The algorithm is halted after a given time limit, which is set to be proportional to the complexity of the instance: indeed, the procedure stops after $\alpha \cdot H \cdot |V| \cdot m$ seconds. We set $\alpha = 1$ for small instances with $H = 3$, $\alpha = 2$ for small instances with $H = 6$, and $\alpha = 0.4$ for large instances. Preliminary experiments showed the quality of the final solution is more dependent on the number of iteration that are performed rather than on the accuracy used for solving **rq-MIRP** at each iteration. For this reason, we impose a time limit equal to $|V| \cdot m/10$ for each solution of model **rq-MIRP**. As to model **AUX**, we always solve it to proven optimally, as the computing time required is very short in this case. In addition, we exploit the fact that the solver may collect alternative optimal solutions; in this case, we use each such solution to produce new routes in both the initialization and diversification phases.

6.3. Comparison with state-of-the-art algorithms

In order to allow for a fair comparison among different algorithms, which were run on different machines and platforms, we standardized the computing times of each method, according to the CPU marks provided by PassMark Single Thread Performance.[1] The computing times of each method, reported in the respective paper, have been multiplied by the mark of the corresponding machine and divided by the mark of the machine where we run ILS-M. Table 1 reports the CPU mark of the machines where the compared algorithms were run.

Our analysis will focus on the average performance of the algorithms across various instances. However, we acknowledge the importance of also considering improved best known bounds on specific instances. Regarding small instances, the recent work by Schenekemberg et al. (2023) has successfully solved to optimality most of them, albeit with significantly larger computational effort compared to any other (exact or heuristic) algorithm. As for large instances, the majority of the best known solutions are provided by the algorithms of either Schenekemberg et al. (2023) or Skålnes et al. (2023). Nevertheless, we have managed to enhance one best solution among the small instances and three among the large instances.

[1] www.cpubenchmark.net/singleThread.html

Table 1
Marks of the CPU used by state-of-the-art methods.

Method	CPU	Mark
AL-SA	Core i7-2600 3.4 GHz	1742
D-M	Intel Core i7-8700K 3.7 GHz	2759
V-M	Xeon Gold 6144 3.5 GHz	2523
SV-BCM	Intel E5-2670v3 2.3 GHz	1691
ILS-M	AMD Ryzen 3960X 3.8 GHz	2676

Table 2
Comparison of the performances of state-of-the-art methods on small instances.

H	m	AL-SA		V-M		D-M		ILS-M		SV-BCM	
		gap	time	gap	time	gap	time	gap	time	gap	time
3	2	0.46	15.42	1.17	14.93	0.04	66.14	**0.00**	168.60	0.01	491.31
	3	0.58	16.06	2.08	16.24	0.12	107.94	**0.03**	252.82	0.03	1754.90
	4	0.56	18.35	1.63	18.65	0.53	155.15	**0.07**	336.57	0.02	2556.21
	5	0.75	19.21	1.50	23.55	0.68	208.17	**0.14**	420.03	0.04	2527.21
6	2	2.08	19.39	0.42	48.83	**0.10**	140.71	0.12	444.43	0.02	2794.18
	3	2.60	19.54	0.42	125.59	0.46	234.23	**0.17**	666.36	0.05	3520.32
	4	2.45	19.54	0.37	167.28	0.81	335.06	**0.21**	888.29	0.05	3805.26
	5	2.92	19.55	0.33	176.25	1.21	426.90	**0.29**	1136.16	0.05	3878.95
		1.30	18.10	1.14	59.66	0.45	189.81	**0.11**	475.93	0.03	2453.18

6.3.1. Analysis of small instances

Table 2 reports the average percentage gap and the average computing time for the algorithms under comparison. For each algorithm, let UB be the upper bound obtained on a given instance. Then, the associated percentage gap is computed as $100(UB - UB^*)/UB^*$, where UB^* denotes the best solution value found among all algorithms in the table, on the same instance. For each row, the lowest average gap among the heuristic methods (leftmost part of the table) is highlighted in bold. Values are aggregated according to network size and inventory cost type, while they are disaggregated in terms of distinct values of the time horizon length (column H) and of the number of vehicles (column m). Thus, each value in the table is an average value over 100 instances in the top part of the table, and over 60 instances in the bottom part. The last row gives average values over all the small instances.

Results in Table 2 show that ILS-M holds its position among state-of-the-art algorithms. Though requiring a larger computing time, on average, it also produces solutions with a much smaller average percentage gap.

Specifically, for $H = 3$, AL-SA outperforms V-M as its average percentage gap is consistently better within similar computing times. Additionally, AL-SA is a preferable choice over D-M for $H = 3$ and $m \geq 4$, as the average gaps of these two algorithms are nearly identical and D-M requires a considerably larger computing time. In this subclass of instances, ILS-M stands out as the only heuristic method that is capable of computing solutions whose quality is comparable to those obtained by SV-BCM, while requiring computing times that are considerably smaller than the exact method. Furthermore, ILS-M dominates SV-BCM when $m \leq 3$.

For $H = 6$, AL-SA is not an attractive option due to its consistently large average percentage gap, though the computing time being very small (at most 20 s). For these instances, V-M exhibits a better performance, outperforming D-M for all values of m but $m = 2$, where D-M takes the lead. For all other combinations of m and $H = 6$, ILS-M consistently achieves the best average percentage gap among the heuristic algorithms.

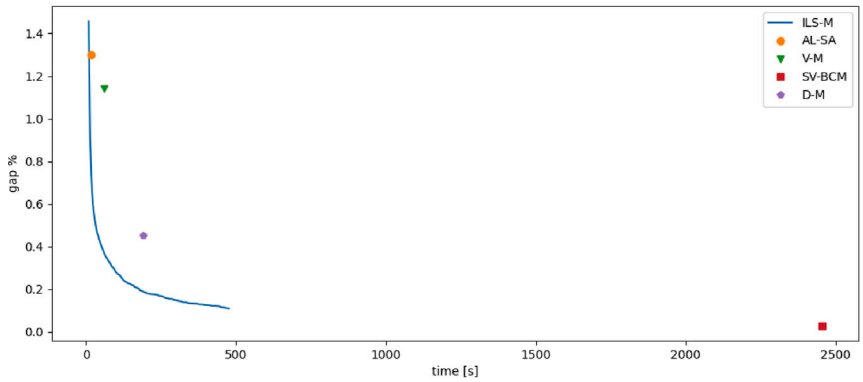


Fig. 1. Average time-gap diagram of state-of-the-art methods on the set of small instances.

Table 3
Comparison of the performances of state-of-the-art methods on large instances.

m	AL-SA		V-M		ILS-M		SV-BCM	
	gap	time	gap	time	gap	time	gap	time
2	5.74	39.33	1.25	329.56	**0.52**	578.41	0.03	4549.40
3	5.54	39.32	0.93	587.15	**0.74**	855.04	0.02	4549.25
4	6.54	39.33	1.07	865.53	**1.02**	1136.08	0.01	4549.41
5	7.06	39.25	**1.25**	1090.56	1.27	1418.81	0.00	4549.28
	6.22	39.31	1.12	718.20	**0.89**	997.08	0.02	4549.34

Fig. 1 plots the average performance of ILS-M, in terms of average percentage gap over all instances, under varying time limits.

The figure clearly shows that our algorithm dominates the other heuristic methods. AL-SA remains competitive for very short computing times only, whereas V-M and D-M are consistently outperformed by ILS-M for average time limits below 500 s. Achieving superior performance requires a considerably larger computational effort, possibly resorting to the use of an exact method as SV-BCM. Therefore, we can assert that, for the category of small instances, ILS-M and SV-BCM are the only two state-of-the-art methods defining the Pareto front in the average time-gap diagram.

6.3.2. Analysis of large instances

Table 3 presents a comparison of the performance of the algorithms on large instances. This table has the same structure as Table 2 but for what concerns the results of algorithm D-M, for which results on large instances were not available in the original paper. In addition, remind that all large instances have $H = 6$ and thus each entry in the table is an average value over 60 instances.

Once again, AL-SA stands out as the fastest method, although the quality of the solutions it produces is quite bad in terms of average percentage gap (more than 6%). For this class of instances, V-M and ILS-M exhibit similar performance in terms of percentage gap and computing time. V-M is less attractive when $m = 2$, with a gap considerably larger than that of ILS-M (1.25 versus 0.52), but it performs better for $m \geq 4$. However, for this class of instances, heuristic methods produce solutions whose associated percentage gap is typically much larger than that of the solutions provided by SV-BCM.

Fig. 2 assists readers in selecting the preferred heuristic method for this instance class. As previously noted, AL-SA fails in computing high-quality solutions, but it is the only algorithm whose running time is very small (less that one minute, on average). For longer average time limits (up to 1000 s), ILS-M turns out to be the best option, as it slightly reduces the average percentage gap with respect to V-M. In this case too, one can obtain better performance at the expense of significantly longer computing times, nearly five times longer, by running SV-BCM.

6.4. The importance of the diversification

One of the most significant and original contributions of our approach is the introduction of model **AUX** for generating useful routes. This model is solved both for initializing the algorithm and for implementing a diversification in the iterated local search procedure. In order to evaluate the effectiveness of this feature, we performed further experiments on all instances using a variant of the algorithm in which the diversification phase is deactivated; the resulting algorithm is referred to as NoDIV hereafter. Furthermore, we also tested intermediate variants where we only apply one of the two objective functions considered for model **AUX** without alternating them (see Section 4.1). The variant employing only the first objective function is denoted as FO1, while FO2 refers to the use of the second objective function only.

Fig. 3 plots the average percentage gap over time for ILS-M and for the three variants of the algorithm we have just introduced. Aside from an initial settling-in period, the two lines corresponding to ILS-M and NoDIV consistently differ by an almost constant term of approximately 0.1%. Eventually, while ILS-M achieves an average percentage gap of 0.32, NoDIV stops at 0.42. On the other hand, FO1 and FO2 exhibit similar behaviors: the respective lines fall within the range between ILS-M and NoDIV, demonstrating that the combination of the two objective functions is effective in diversifying the exploration of the solution space.

6.5. Column generation strategies

In Section 5.2 we introduced three different strategies to generate new columns during the execution of the local search procedure. We now analyze the effect of each strategy on the performance of the algorithm. In Fig. 4, we compare algorithm ILS-M with three variants: (i) NoCVRP, where we deactivate the first strategy, asking for the solution of a CVRP instance for each period; (ii) NoCopy, where we deactivate the second strategy, based on the copy of routes across time periods; and (iii) NoReal, where we deactivate the third strategy, which reallocates customers among routes (see Algorithm 2).

The figure clearly shows that the third strategy is crucial for the effectiveness of the algorithm: by removing this feature, the average percentage gap significantly worsens and is always larger than 1.2, even for large time limits. The impact of the second strategy on the overall performances is less evident, though being still non-negligible. Except for an initial time lapse, NoCVRP consistently differs from ILS-M by more than 0.1%. Finally, NoCopy is the only variant yielding an average percentage gap that is comparable to ILS-M, producing the best performance when average time limit is very short (less than 200 s); however, for larger time limits, ILS-M remains the most effective strategy.

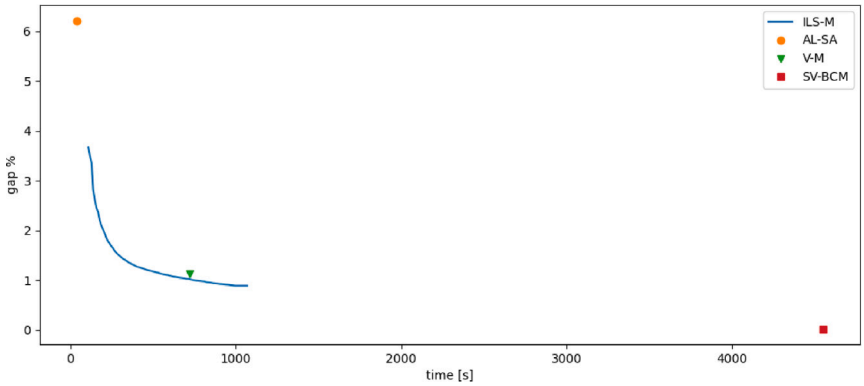


Fig. 2. Average time-gap diagram of state-of-the-art methods on the set of large instances.

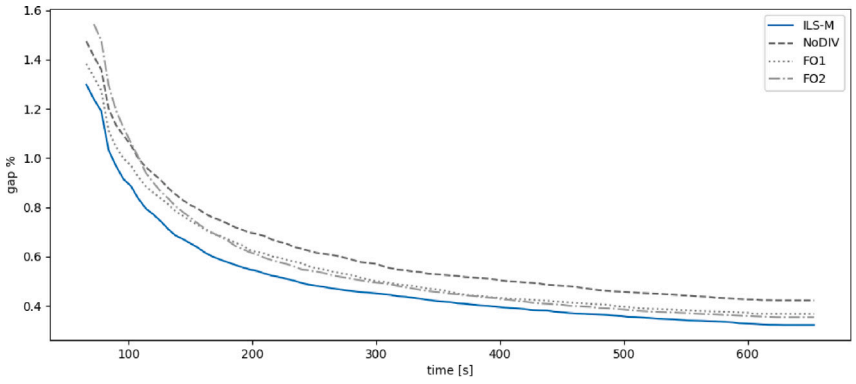


Fig. 3. Comparison between ILS-M and variants with different diversification strategies (all instances).

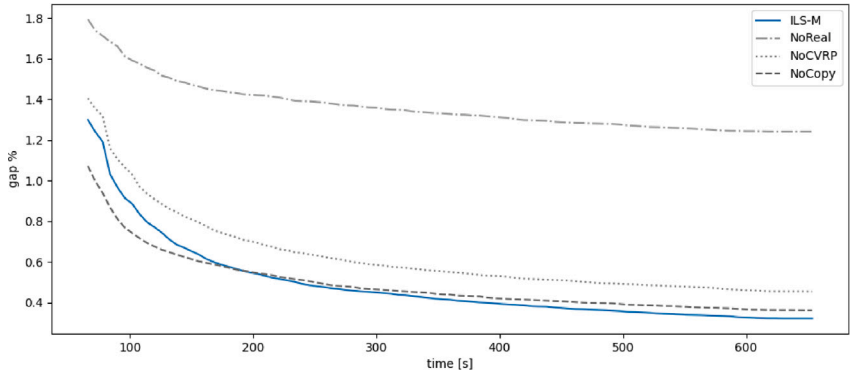


Fig. 4. Comparison between ILS-M and variants with different column generation strategies (all instances).

6.6. Analysis of the results on DIMACS challenge instances

In 2020 the 12th DIMACS Implementation Challenge was launched, with the aim of developing and implementing algorithms for Vehicle Routing problems (broadly defined), including **MIRP**. The Challenge terminated in 2022 and included a benchmark of instances and a reference computational setting. Although derived from those in the existing literature, the new instances differ from the previous ones for some aspects, including vehicle capacity and inventory cost at the depot, which prevents a direct comparison of the two benchmark sets. In addition, the DIMACS Challenge introduced new instances with $|V| \in \{35, 40, 45, 50\}$ and $H = 6$. Overall, the DIMACS benchmark includes 1038 instances.

The computational setting prescribed for the Challenge imposed to test the algorithms with a fixed time limit on a specific hardware. This time could be scaled on different machines according to the PassMark Single Thread Performance as discussed in Section 6.3, and is identical for all instances, regardless their size. In our computational environment, this resulted in a time limit equal to 1340 s per instance.

We evaluated our algorithm on the DIMACS instances under the same setting imposed by the Challenge, and report a comparison with all the participants to the Challenge for **MIRP** in Table 4. The column headers in Table 4 refer to the algorithms competing in the Challenge. For each algorithm we give the average percentage gap, computed with respect to the best solution found by the considered methods. We partitioned the DIMACS benchmark according to the size of instances and the number H of periods. The third column (#inst) reports the number of instances in each group. All the instances and the detailed numerical results from the competition are available on the website of the Challenge http://dimacs.rutgers.edu/programs/challenge/vrp/.

Table 4
Comparison of the performances of the average percentage gap on DIMACS Challenge instances.

Class	H	#inst	plaisir	TSMHA	IRPUC	2FHBC	MrOptimal	ILS-M
Small	3	400	2.37	0.06	0.11	0.05	0.09	**0.04**
	6	398	4.56	0.26	0.45	**0.16**	0.27	0.48
Large	6	240	22.47	1.35	1.62	1.20	**0.11**	0.74

The table shows that, though designed for a different setting in which a considerably smaller time limit is allowed, ILS-M turns out to be the best algorithm on instances with $H = 3$ and the second best for large instances.

6.7. Computational experiments with the OU policy

In this section, we aim to assess the versatility of our method by adapting it to another well-known version of the problem that imposes a more restrictive OU policy. The most recent works attacking this problem are:

- Adulyasak et al. (2014) who proposed both an exact approach based on the solution of three MILPs and an adaptive large neighborhood search (ALNS) heuristic algorithm;
- Zhang et al. (2021) who designed an algorithm based on a Benders decomposition approach;
- Solyalı and Süral (2022) who also adapted the proposed matheuristic algorithm, originally designed for the problem with the ML policy (MATHIRP), to this policy.

The authors of the last and most recent paper conducted an extensive comparison of all the aforementioned methods, despite encountering some discrepancies. Initially, the benchmark comprised 300 instances, as proposed by Adulyasak et al. (2014). However, it has recently been expanded to include an additional 20 instances by Zhang et al. (2021); on the other hand, the algorithm of Zhang et al. (2021) fails to find a primal bound in several instances. In addition, the primal bounds found by Adulyasak et al. (2014) are, actually, the best among three different approaches: two of them run as a single-thread with a two-hour time limit and the remaining one employs eight threads and has a time limit of 12 h. Hence, evaluating the computing time of this algorithm is hard, given its highly time-consuming nature. These discrepancies also complicate the calculation of fair average values using a standardized benchmark. We can summarize the computational analysis in Solyalı and Süral (2022) by highlighting that their MATHIRP algorithm outperforms the ALNS heuristic in terms of solution quality and, on average, competes effectively with the two exact methods. Specifically, the exact methods turns out to be superior when small instances are addressed, but yields poor performances on larger instances. For further details, readers are referred to Solyalı and Süral (2022). In our analysis, we will focus on comparing our approach with MATHIRP, not only due to the aforementioned issues but also because of its similarity in settings, making it our most direct competitor.

6.7.1. Instances for OU policy

The benchmark of instances used for this problem version is a subset of the one considered in the previous section with the ML policy. In order to describe this benchmark, we find it easier to use Table 5, which provides a breakdown of instances for each combination of time-horizon length (column H), the number of vehicles (column m), and inventory cost type (third column). Columns from the fourth to the last distinguish the various values of the cardinality of the vertices ($|V|$). In each cell containing a bullet, there are five instances.

6.7.2. Analysis of the results with OU policy

Table 6 follows the same structure as Table 2, and presents the average percentage gap with respect to the best solution found by the two compared methods. So as to have a fair comparison, computing times of MATHIRP have been scaled according to the CPU mark (Xeon X5650 2.67 GHz, 1300) used in this analysis.

Table 5
Number of instances used as benchmark for the OU version of the problem.

H	m		5	10	15	20	25	30	35	40	45	50
3	2	High	•	•	•	•	•					
		Low	•	•	•	•	•					
	3	High	•	•	•	•	•	•	•	•	•	•
		Low	•	•	•	•	•	•	•	•	•	•
	4	High						•	•	•	•	•
		Low						•	•	•	•	•
6	2	High	•	•	•	•	•					
		Low	•	•	•	•	•					
	3	High	•	•	•	•	•	•				
		Low	•	•	•	•	•	•				
	4	High						•				
		Low						•				

Table 6
Comparison of the performances of state-of-the-art methods with OU policy.

H	m	MATHIRP		ILS-M	
		gap	time	gap	time
3	2	0.38	2.69	**0.10**	90.21
	3	0.37	6.46	**0.32**	251.75
	4	**0.33**	105.16	0.67	492.55
6	2	0.61	19.20	**0.17**	384.55
	3	0.61	1257.79	**0.37**	646.33
	4	0.78	1775.73	**0.01**	1488.98
		0.46	313.20	**0.32**	397.53

The table shows that, on all instances but those with $H = 3$ and $m = 4$, ILS-M computes solutions whose quality is better than those produced by MATHIRP. In addition, ILS-M significantly outperforms MATHIRP on the most challenging instances, namely those with $H = 6$ and $m \geq 3$. Similarly, ILS-M obtains better solutions on instances with $H = 6$ and $m = 2$ and on those with $H = 3$ and $m \leq 3$, although at the expense of larger computing times. Among all cases, MATHIRP dominates ILS-M for the case $H = 3$ and $m = 4$ only. The last row of the table shows that, overall, ILS-M finds better solutions than MATHIRP at the expense of a slightly longer computing time, thus confirming that the proposed approach is the preferable choice on this variant of the problem.

7. Conclusions

We proposed an effective matheuristic algorithm for the **MIRP**. Our algorithm is built on top of a new mathematical formulation in which, besides inventory and delivery decisions, variables are associated with routes. As the formulation includes an exponential number of such variables, we use an iterative approach in which the set of route-variables in the current formulation is dynamically updated at each iteration. In doing so, we take care of defining, at each iteration, a formulation whose size remains manageable, so as to allow its efficient solution. A diversification phase comes into play whenever a local optimum is reached. Computational results on benchmark instances with a MP policy show that the proposed matheuristic is competitive with state-of-the-art heuristic algorithms in terms of Pareto optimality with respect to the average solution quality and computational effort. In particular, on small instances, our algorithms outperforms all the existing heuristics and competes effectively with the best sequential exact algorithm proposed so far in the literature for instances with 3 time periods. On large instances, our algorithm performs better than the other heuristics from the literature, provided a sufficiently large computing time is available. We compared the performance of our algorithm on the so-called OU policy as well, showing that our

algorithm performs on average better than the best competitor from the literature in this case as well.

The adopted formulation seems to be suitable for attacking other routing problems in which tactical decisions are embedded (e.g., location routing and production routing problems), which suggests the adoption of a similar approach for these problems. In addition, a future research direction could investigate the integration of machine learning techniques in the route generation phase. These techniques could be beneficial in determining near-optimal clustering of the customers within the planning horizon, by taking into account their main features, e.g., demand levels, maximum inventory constraints, and distance.

CRediT authorship contribution statement

Demetrio Laganà: Conceptualization, Formal analysis, Investigation, Validation, Writing – original draft, Writing – review & editing, Methodology, Supervision. **Enrico Malaguti:** Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Validation, Writing – original draft, Writing – review & editing. **Michele Monaci:** Conceptualization, Formal analysis, Investigation, Methodology, Supervision, Validation, Writing – original draft, Writing – review & editing. **Roberto Musmanno:** Conceptualization, Project administration, Resources, Validation, Writing – original draft, Writing – review & editing. **Paolo Paronuzzi:** Writing – original draft, Writing – review & editing, Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation.

Data availability

Data will be made available on request.

Acknowledgments

The authors are grateful to three anonymous reviewers for their constructive comments and remarks.

The work of Enrico Malaguti, Michele Monaci and Paolo Paronuzzi was partially funded by PRIN project “Large-scale optimization for sustainable and resilient energy systems” and by the University of Calabria, through project grant CIG8929018ABE.

References

Adulyasak, Y., Cordeau, J.F., Jans, R., 2014. Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. INFORMS J. Comput. 26 (1), 103–120.

Alvarez, A., Munari, P., Morabito, R., 2018. Iterated local search and simulated annealing algorithms for the inventory routing problem. Int. Trans. Oper. Res. 25 (6), 1785–1809.

Andersson, H., Hoff, A., Christiansen, M., Hasle, G., Løkketangen, A., 2010. Industrial aspects and literature survey: Combined inventory management and routing. Comput. Oper. Res. 37 (9), 1515–1536.

Applegate, D.L., Bixby, R.E., Chvatal, V., Cook, W.J., 2007. The Traveling Salesman Problem: A Computational Study. Princeton University Press, USA, ISBN: 0691129932.

Archetti, C., Bertazzi, L., 2021. Recent challenges in routing and inventory routing: E–commerce and last–mile delivery. Networks 77 (2), 255–268.

Archetti, C., Bertazzi, L., Hertz, A., Speranza, M.G., 2012. A hybrid heuristic for an inventory routing problem. INFORMS J. Comput. 24 (1), 101–116.

Archetti, C., Bertazzi, L., Laporte, G., Speranza, M.G., 2007. A branch-and-cut algorithm for a vendor-managed inventory-routing problem. Transp. Sci. 41, 382–391.

Archetti, C., Bianchessi, N., Irnich, S., Speranza, M.G., 2014. Formulations for an inventory routing problem. Int. Trans. Oper. Res. 21 (3), 353–374.

Archetti, C., Boland, N., Speranza, M.G., 2017. A matheuristic for the multivehicle inventory routing problem. INFORMS J. Comput. 29 (3), 377–387.

Archetti, C., Guastaroba, G., Huerta-Muñoz, D.L., Speranza, M.G., 2021. A kernel search heuristic for the multivehicle inventory routing problem. Int. Trans. Oper. Res. 28 (6), 2984–3013.

Archetti, C., Speranza, M.G., 2016. The inventory routing problem: The value of integration. Int. Trans. Oper. Res. 23 (3), 393–407.

Bell, W.J., Dalberto, L.M., Fisher, M.L., Greenfield, A.J., Jaikumar, R., Kedia, P., Mack, R.G., Prutzman, P.J., 1983. Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. Interfaces 13 (6), 4–23.

Bertazzi, L., Coelho, L.C., De Maio, A., Laganá, D., 2019. A matheuristic algorithm for the multi-depot inventory routing problem. Transp. Res. Part E: Logist. Transp. Rev. 122, 524–544.

Bertazzi, L., Savelsbergh, M., Speranza, M.G., 2008. Inventory routing. In: The Vehicle Routing Problem: Latest Advances and New Challenges. Springer US, Boston, MA, pp. 49–72.

Bertazzi, L., Speranza, M.G., 2012. Inventory routing problems: An introduction. EURO J. Transp. Logist. 1, 307–326.

Bertazzi, L., Speranza, M.G., 2013. Inventory routing problems with multiple customers. EURO J. Transp. Logist. 2, 255–275.

Campbell, A., Clarke, L., Kleywegt, A., Savelsbergh, M., 1998. The inventory routing problem. In: Fleet Management and Logistics. Springer US, Boston, MA, pp. 95–113.

Čankaya, E., Ekici, A., Özener, O.Ö., 2019. Humanitarian relief supplies distribution: An application of inventory routing problem. Ann. Oper. Res. 283, 119–141.

Cárdenas-Barrón, L.E., Melo, R.A., 2021. A fast and effective MIP-based heuristic for a selective and periodic inventory routing problem in reverse logistics. Omega 103, 102394.

Chitsaz, M., Cordeau, J.F., Jans, R., 2019. A unified decomposition matheuristic for assembly, production, and inventory routing. INFORMS J. Comput. 31 (1), 134–152.

Christiansen, M., Fagerholt, K., Flatberg, T., Haugen, Ø., Kloster, O., Lund, E.H., 2011. Maritime inventory routing with multiple products: A case study from the cement industry. European J. Oper. Res. 208, 86–94.

Coelho, L.C., Cordeau, J.F., Laporte, G., 2012. Consistency in multi-vehicle inventory-routing. Transp. Res. C 24, 270–287.

Coelho, L.C., Cordeau, J.F., Laporte, G., 2014. Thirty years of inventory routing. Transp. Sci. 48 (1), 1–19.

Coelho, L.C., Laporte, G., 2013. The exact solution of several classes of inventory-routing problems. Comput. Oper. Res. 40 (2), 558–565.

Desaulniers, G., Rakke, J.G., Coelho, L.C., 2016. A branch-price-and-cut algorithm for the inventory-routing problem. Transp. Sci. 50 (3), 1060–1076.

Diniz, P., Martinelli, R., Poggi, M., 2020. An efficient matheuristic for the inventory routing problem. In: Combinatorial Optimization: 6th International Symposium, ISCO 2020, Montreal, QC, Canada, May 4–6, 2020, Revised Selected Papers 6. Springer, pp. 273–285.

Dušan, H., Hvattum, L.M., Hoff, A., 2022. The value of integrated planning for production, inventory, and routing decisions: A systematic review and meta-analysis. Int. J. Prod. Econ. 248, 108468.

Gaur, V., Fisher, M.L., 2004. A periodic inventory routing problem at a supermarket chain. Oper. Res. 52, 813–822.

Guimarães, T.A., Schenekemberg, C.M., Coelho, L.C., Scarpin, C.T., Pécora Jr., J.E., 2023. Mechanisms for feasibility and improvement for inventory-routing problems. J. Oper. Res. Soc. 1–13.

Laganá, D., Longo, F., Vocaturo, F., 2016. Vendor-managed inventory practice in the supermarket supply chain. Int. J. Food Eng. 12, 827–834.

Manousakis, E., Repoussis, P., Zachariadis, E., Tarantilis, C., 2021. Improved branch-and-cut for the inventory routing problem based on a two-commodity flow formulation. European J. Oper. Res. 290 (3), 870–885.

McKenna, R.S., Robbins, M.J., Lunday, B.J.I.M.M., 2020. Approximate dynamic programming for the military inventory routing problem. Ann. Oper. Res. 288, 391–416.

Moin, N.H., Salhi, S., 2007. Inventory routing problems: A logistical overview. J. Oper. Res. Soc. 58 (9), 1185–1194.

Moin, N.H., Salhi, S., Aziz, N.A.B., 2011. An efficient hybrid genetic algorithm for the multi-product multi-period inventory routing problem. Int. J. Prod. Econ. 133 (1), 334–343.

Popović, D., Vidović, M., Radivojević, G., 2012. Variable neighborhood search heuristic for the inventory routing problem in fuel delivery". Expert Syst. Appl. 39, 13390–13398.

Santos, E., Ochi, L.S., Simonetti, L., González, P.H., 2016. A hybrid heuristic based on iterated local search for multivehicle inventory routing problem. Electron. Notes Discrete Math. 52, 197–204.

Schenekemberg, C., Guimarães, T., Chaves, A., Coelho, L., 2023. A three–front parallel branch-and-cut algorithm for production and inventory routing problems. Transp. Sci..

Skålnes, J., Vadseth, S., Andersson, H., Stålhane, M., 2023. A branch-and-cut embedded matheuristic for the inventory routing problem. Comput. Oper. Res. 159, 106353.

Solyalı, O., Süral, H., 2011. A branch-and-cut algorithm using a strong formulation and an a priori tour-based heuristic for an inventory-routing problem. Transp. Sci. 45 (3), 335–345.

Solyalı, O., Süral, H., 2017. A multi-phase heuristic for the production routing problem. Comput. Oper. Res. 87, 114–124.

Solyalı, O., Süral, H., 2022. An effective matheuristic for the multivehicle inventory routing problem. Transp. Sci. 56 (4), 1044–1057.
Tinič, G.Ö., Koca, E., Yaman, H., 2021. An exact solution approach for the inventory routing problem with time windows. Comput. Oper. Res. 134, 105371.
Vadseth, S.T., Andersson, H., Stålhane, M., 2021. An iterative matheuristic for the inventory routing problem. Comput. Oper. Res. 131, 105262.
Vidal, T., 2022. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. Comput. Oper. Res. 140, 105643.
Vincent, F.Y., Widjaja, A.T., Gunawan, A., Vansteenwegen, P., 2021. The multi-vehicle cyclic inventory routing problem: Formulation and a metaheuristic approach. Comput. Ind. Eng. 157, 107320.
Zhang, Z., Luo, Z., Baldacci, R., Lim, A., 2021. A benders decomposition approach for the multivehicle production routing problem with order-up-to-level policy. Transp. Sci. 55 (1), 160–178.