Electronic Research Archive

ERA, 34(12): 9386–9405.
DOI: 10.3934/era.2026394
Received: 24 February 2026
Revised: 30 April 2026
Accepted: 03 July 2026
Published: 10 November 2026

http://www.aimspress.com/journal/era

*Research article*

# Solving the inventory routing problem with the Archimedes optimization algorithm

**Kevin Neuschwander and Thomas Hanne***

School of Business, University of Applied Sciences and Arts Northwestern Switzerland, Riggenbachstrasse 16, Olten 4600, Switzerland

* **Correspondence:** Email: thomas.hanne@fhnw.ch.

**Abstract:** The inventory routing problem is a frequently occurring optimization problem in logistics processes, which considers scenarios where a vendor must find optimal solutions to distribute goods to multiple customers while having some flexibility regarding shipping times (i.e., on which days which quantities are delivered to a customer) due to warehousing possibilities. Due to the complexity of the problem, various optimization algorithms have been applied to the problem. As there is still potential for further improvements, we investigate whether the Archimedes optimization algorithm, a rather novel metaheuristic approach, can perform better compared to established algorithms such as genetic algorithms or ant colony optimization. The results show that the Archimedes optimization algorithm can be adapted and successfully applied to both basic cases of the inventory routing problem and more complex cases in small and large scenarios. In the multi-depot IRP, the algorithm finds better solutions than a genetic algorithm, which can be attributed to the fact that the genetic algorithm makes smaller changes, and more solutions must be generated to explore the solution space. However, these smaller changes are faster and allow the genetic algorithm to fine-tune solutions that are close to a local or global optimum. In the multi-product inventory routing problem, the Archimedes optimization algorithm is compared with ant colony optimization. The more complex route determination logic makes ant colony optimization slower than the Archimedes optimization algorithm but allows ant colony optimization to achieve better routes. However, ant colony optimization cannot define the delivery quantities very well, which is better achieved by the Archimedes optimization algorithm. Using the advantages of all three algorithms and generating a combined algorithm may help find even better solutions to inventory routing problems.

**Keywords:** inventory routing problem; metaheuristics; Archimedes optimization algorithm; ant colony optimization; genetic algorithm; logistics

## 1. Introduction

The inventory routing problem (IRP) is an important optimization problem that has been researched since the 1980s. This problem is relevant for all vendors that distribute goods to customers if there is some flexibility regarding transport time due to utilizable warehouse capacities, such as in vendor managed inventory agreements [1]. Every customer has specific requirements, such as when to deliver the product or how much is required. In addition, there are constraints, such as the inventory size. The vendor must fill the customer's inventory and ensure that the stock never falls below a certain level. This must be accomplished by considering the limited number of resources [2]. The delivery part of the inventory routing problem corresponds to the well-known vehicle routing problem (VRP). Bertazzi and Speranza [1] remark that this optimization problem is the most complex part of an IRP. An automatic algorithm that defines the necessary routes a truck must take and the quantities that must be delivered to each customer can help a vendor reduce costs by saving time to manually find a solution [3].

The IRP is a computationally complex optimization problem involving many variables and various constraints, as the problem involves a variety of subproblems, such as the VRP and inventory management problems. Shaabani [3] explained that combining multiple problems better reflects real-life situations and, therefore, helps find a better solution for the entire organization and its stakeholders. However, according to [4], the problem can become a very complex task to manage for certain organizations, such as retailers, where many products are to be provided at many different locations, such as in a so-called Multi-Product IRP (MPIRP) scenario. Alternatively, a vendor could have products available at different locations and must find a way to optimally deliver these products to the customer as part of a Multi-Depot IRP (MDIRP). Such a problem is considered, for instance, in [5], where the depots are denoted as warehouses and themselves part of the problem solutions as a location-IRP is studied. According to [6], not only physical constraints, such as transport or warehousing capacities, are possible in the IRP. In some cases, it is also the time when actions, such as deliveries, need to be executed that must be considered by the algorithm, which further increases the complexity [6].

Finding an optimal solution to the IRP can help logistics companies and departments with limited resources reduce costs and increase customer satisfaction. According to Shaabani [3], it is a critical success factor for an organization handling perishable materials to have an optimized solution to the problem. A suboptimal solution may lead to a decomposition of materials before they are consumed [3]. Nikzad et al. [7] stated that finding a solution to this problem is also crucial for the distribution of drugs. Organizations must ensure that sufficient drugs are available at any time in hospitals and other healthcare institutions [7]. Malladi and Sowlati [8] outlined that algorithms can also help organizations become more sustainable. An optimized method can help to minimize waste production and gas consumption [8]. According to Archetti and Bertazzi [9], algorithms that re-solve inventory routing problems can also help e-commerce markets, which often receive a large number of online orders from many customers. These customers expect a fast delivery of goods and often deliveries at a specific time of the day. Companies could introduce new delivery models in which drivers are hired to drive a

specific route to serve relevant customers. In addition, drones can also be used to deliver orders. To determine routes for drivers or drones, an algorithm could be used, which could significantly reduce the costs for the company [9].

Existing solution approaches either do not consider all variables and constraints or require significant time and computational resources to find a solution. While the routing part of an IRP (even for a single-product, single-depot version) is already NP-hard, as it is a variant of the travelling salesman problem, Baller et al. [10] showed that IRP variants, even with a strongly simplified topology, can become NP-hard due to the temporal planning task. Thus, the runtime for solving the problem may exponentially increase with the problem size depending, e.g., on the number of customers to be served or the time horizon [10]. Çelik et al. [11] mentioned that due to the complexity of the problem, many heuristic and metaheuristic approaches have been developed besides exact methods, which aim at optimal solutions but would require more processing time. Saragih et al. [12] mentioned that larger-scale problems are becoming too complex for exact methods to solve. Therefore, there is an ongoing need to find efficient optimization approaches. In particular, for specific variants of the IRP such as with multiple products and multiple depots (see below), there is little evidence of the comparative performance of state-of-the-art metaheuristics.

In recent years, new optimization algorithms have been developed that can significantly improve the performance of more established metaheuristics. For many of these algorithms, there are no computational results available regarding the possible application to the IRP. One of those algorithms is the Archimedes optimization algorithm (AOA) [13]. In particular, this algorithm has evoked academic interest leading to further analyses, suggestions for improvements, and various successful applications [14]. However, the AOA has not yet been applied to any variant of the IRP. Our hypothesis is that the AOA can find better solutions to the MDIRP and the MPIRP than existing metaheuristic optimization approaches, such as the ant colony optimization (ACO) algorithm or a genetic algorithm (GA), in a scenario where a supplier with capacitated production and logistics resources must deliver goods to many customers. GAs are based on the idea to simulate genetic evolution by representing solutions as genes during an optimization process which goes through various steps during a number of iterations (generations) such as replication with mutations and crossover and selection making use of fitness evaluation making based on the objective function. ACO is a swarm-based algorithm which simulates the food search process of ants making use of a probabilistic path construction considering the desirability of edges and pheromones left from other ants during previous search. Pheromones are considered as a communication means in solution construction to favor better solutions on the one hand (adding pheromone based on solution quality) and evaporation of pheromones on the other hand to avoid getting stuck in the local optima.

The two approaches, GA and ACO, are selected for comparison, as those are well-established metaheuristics and have been effectively used to solve variants of the IRP and similar problems. In addition, all three methods follow a quite different algorithmic logic, which may provide further insights into their mutual merits. In our study, we consider the following main research question: Does the AOA find better solutions to IRPs than ACO or a GA? One main challenge for answering this question is the adoption of the algorithm to the specific optimization problem and its constraints in order to run efficiently. Thus, the main question is broken down into the following three sub-research questions: 1) How can the AOA be applied to IRPs? 2) Does the AOA have a faster runtime compared with ACO and a GA? 3) Are solutions proposed by the AOA better in terms of objective values (i.e., costs of found solutions) than those proposed by ACO or a GA?

The main contributions of our paper are as follows: 1) We show how AOA can be applied to 2 variants of the IRP with different complexity. 2)We suggest an experimental framework to compare AOA with a GA and ACO, including the generation of test instances and the specification of the evaluation criteria, and based on this, we conduct experiments in order to evaluate and compare the performance of the considered algorithms. 3) Insights into the studied algorithms and their application to the IRP may help managers to better solve respective planning scenarios in practice.

Our paper is structured as follows: In Section 2, we discuss the background of the IRP; then, approaches to solve the IRP such as ACO, the GA, and the AOA are discussed in Section 3; the setups of the computational experiments to evaluate the AOA compared with a GA and ACO are described in Section 4, together with the respective results; and in Section 5, our conclusions are presented.

## 2. Background on the IRP

As explained above, the IRP is basically a combination of a VRP and an inventory planning problem. As in the VRP, optimal routes must be found to serve a number of customers using trucks with limited transport capacities. However, due to inventories and available warehouse capacities, there is some flexibility when particular customers are being served.

Figure 1 illustrates a simple instance of the VRP part. The vendor has a depot which is represented by a factory icon on the left-hand side of the figure. The numbered circles act as the customers of this vendor. The lines illustrate the paths between the different nodes. Each path has a cost which is represented by the number written next to it. There are a variety of methods for how these costs can be calculated or determined, such as the time needed to travel from one node to the other.

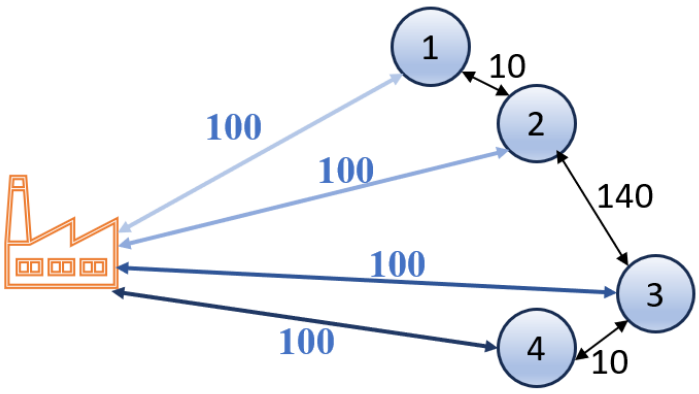



**Figure 1**. Vehicle Routing Problem Example based on [1].

In the simplest case, a vehicle could travel to customer 1, unload, and return to the depot. The cost would be 200. If the same was done for customer 2, the cost of this second trip would be again 200, which would yield a total cost of 400. A truck with enough capacity could serve customers 1 and 2 at the same time. This would reduce the costs from 400 to 210. The goal of an algorithm dealing with the problem is to find a path with the lowest costs.

In case of the IRP, more constraints, such as inventory size or inventory storage costs, are to be considered in the calculation, and the overall goal is cost minimization, which also includes inventory

costs. In Figure 1 each node is assigned to specific constraints: The depot node has a specific production per time unit, and several trucks which can leave the depot in one time unit. Additionally, each of these trucks has a capacity t, which specifies how much of the product can be transported. A customer node has an inventory level, a maximum capacity, and a consumption. In addition, a constraint defining how many deliveries can be made in one time unit is introduced. Contrary to the VRP, not every customer needs to be visited every day. However, the supplier must assure that the customers are served with enough product, including their inventory to satisfy the consumption. In addition, it is not possible to deliver more than the maximum inventory capacity. For the inventory levels described in the figure, nodes 2 and 3 must be served first, to avoid an interruption in production. Node 3 can only be served by one truck per time unit. This means that the best option could be that one of the trucks from the depot visits node 3, fully unloads, and returns to the depot. The other two trucks could satisfy the capacity of node 2 and unload the rest in either node 1 or node 4.

Meanwhile, many different formulations of the IRP have been discussed. Archetti and Ljubić [15] provided an overview of some respective IRP variants. Additionally, problem variants including uncertainty have been recently investigated [16,17]. [16,17] considered uncertain demand at the retailers which may result in uncertain inventory during the planning horizon.

Another possible extension of IRP formulations is to consider multiple objectives [18,19]. In [18], economic goals (e.g., minimizing network cost), environmental goals (e.g., minimizing emission penalty), and social goals (e.g., maximizing supplier incentives) were discussed, while [19] discussed an approach to minimize transportation costs and greenhouse gas emissions. Occasionally, real-world cases were also studied, such as in [20]. We would also like to mention that the even more complex problem of location-inventory-routing, which includes the problem of specifying facility or warehouse locations, has occasionally been investigated [21–23].

As explained in Section 4 in more detail, for our study, we investigate two setups of the IRP, a basic setup which considers a single product, a single depot, and homogenous trucks (with identical capacity). Warehouse capacity limits are not considered here, and product consumption is constant during the planning horizon. In an extended setup, we consider multiple products, multiple depots, heterogeneous trucks, limited warehouse capacities, and dynamic product consumption.

## 3. Algorithms for the IRP

The solution approaches to the IRPs have been intensively studied in the scientific literature. According to [24], there exist exact methods that guarantee to find the optimal solution or heuristics that attempt to approximate an optimal solution. Shaabani [3] adds that metaheuristics, matheuristics, and hybrid methods are other approaches that are used to address the IRP. Further investigations on matheuristics were contributed in [25–27]. Some of the approaches are hybrid approaches and further hybrid approaches are, e.g., treated in [28].

Exact methods were found suitable for smaller scenarios; however, they encounter problems when dealing with larger scenarios with many customers. Therefore, our study focuses on metaheuristic methods which do not necessarily find the best solutions but can also handle the bigger scenarios with reasonable computational time requirements. In the following, we discuss two established methods for the IRP and a novel one, which is further explored in our study. Let us note that some other rather novel metaheuristics, such as the crow search algorithm [29], the firefly algorithm [30], or the whale optimization algorithm [23], have been recently investigated to solve an IRP.

### *3.1. Ant colony optimization*

Wong and Moin [31] described an ant colony algorithm that simulates the behavior of an ant colony that spreads pheromones to guide an ant to places of interest previously found by other ants. In the case of an IRP, this concept is used to find optimal routes for trucks. If a low-cost solution is found, then this increases the pheromone concentration on the route segments used in this solution. Together with the path length of a segment, this pheromone concentration is used to determine the value of the segment. The higher the value, the greater is the probability that this path segment will be used in the adapted solution. This way, the algorithm converges by continuously including the most efficient segments found so far. By not always using the most efficient route, the algorithm also ensures that alternative routes are tested. Because the pheromone concentration is lower in earlier iterations, alternative routes are chosen more frequently than in later iterations where the pheromone concentration could lead to a path value much higher than the path values of the alternative routes. This allows the algorithm to first scan a larger area in the solution space and, in the end, perform a more detailed local search in the area of the best solution found so far. The authors use this logic to find optimal routes for the MPIRP [31].

### *3.2. Genetic algorithm*

Hiassat et al. [5] used a GA in a MDIRP. GAs imitate the biological processes of evolution and reproduction. During reproduction, genes of the parents are recombined and transmitted to the children. In addition, mutations can cause unexpected changes in genes and, therefore, in offspring solutions. In the end, the best genes survive because they have an advantage over others. GAs use this principle to solve optimization problems, such as the IRP. Solutions are represented by a gene-like structure. The gene in the GA is a one-dimensional sequence of numbers. This sequence contains the customers to be served by a specific depot and the product quantities that these customers receive. While the quantity is already numeric, an ID number must be assigned to each customer to identify them in the gene. The sequence in which customers appear in the gene defines the sequence in which these customers are served and thus the transport route. In each iteration, offspring solutions are generated by mutating the parent solutions. These mutations include swapping customers, thus changing the transport routes and swapping the delivery quantities. Alternatively, the routes and quantities of two parent solutions can be combined to form an offspring solution. A simple combination of two parent solutions is to take the routes of time period 1 from parent solution A and the routes of time period 2 from parent solution B. From the parent and offspring solutions, the best solutions are selected and used in the next iteration until the maximum number of iterations has been reached. Finally, the best-found solution is returned by the algorithm [5].

### *3.3. The Archimedes optimization algorithm*

Hashim et al. [13] introduced a new optimization algorithm called the AOA. This algorithm attempts to solve optimization problems by simulating the behavior of multiple objects that are thrown into a liquid at the same time and attempts to reach the equilibrium state. Each of these objects represents a candidate solution. This makes it a population-based metaheuristic optimization method.

The algorithm begins by generating the first solutions. Each solution has a density, a volume, and an acceleration property. In a real scenario where objects are thrown into a liquid, these properties affect the equilibrium state of the objects in the liquid. In the simulation, these properties control how the solution is updated. In an iteration of the algorithm, these properties are updated as a first step. The density and volume are updated based on the best solution found so far. The acceleration update is based on the volume and density of the currently processed solution and a randomly selected other solution. The more different these properties are, the larger is the update. After updating the acceleration, the acceleration is normalized to prevent this variable from indefinitely growing. The normalized acceleration $a^{t+1}$ for iteration $t$+1 is used in Eqs (1) and (2), which update the vector variables (solution $i$ in iteration $t$ is $p_i^t$) of the solution with components representing specific aspects of a solution (e.g., a quantity received by a customer in a specific period) [13]:

$$p_i^{t+1}=p_i^t+x*r*\ a_i^{t+1}*td*(p_o-p_i^t), \tag{1}$$

$$p_i^{t+1}=p_{best}^t+F*y*r*\ a_i^{t+1}*td*(2*tf*p_{best}-p_i^t). \tag{2}$$

The transfer factor (*tf*) and the changing degree (*td*) are two more important properties that affect how the solution is updated. The transfer factor is calculated as $tf=\exp\left(\frac{t-t_{max}}{t_{max}}\right)$ (i.e., it is based on the current iteration $t$ in relation to the maximum number of iterations $t_{max}$) and controls which update equation is used. Equation (1) is used in earlier iterations (when $tf \leq 0.5$), and Eq (2) is used in later iterations. The changing degree decreases with each iteration, which leads to smaller changes in later iterations. It is calculated as $td=\exp\left(\frac{t-t_{max}}{t_{max}}\right)-\frac{t}{t_{max}}$.

$x$ and $y$ are two parameters that contribute to the solution update. A high $x$ value leads to large changes in the solution in earlier iterations, whereas a high $y$ value updates solutions faster in later iterations. The authors of the algorithm set these values to 2 and 6, respectively. In addition, the solution update is controlled by a random variable ($r$) between 0 and 1. In addition, the current solution values of the solution ($p_i^t$) and, in the case of Equation 1, the values of a random interacting solution ($p_o$), are considered. In Eq (2), the interacting solution is always the best solution found so far ($p_{best}$). For later iterations in which Eq (2) is used, the F-Variable ($F$) is important, which controls whether the updated solution moves toward the best solution found thus far or moves in the opposite direction. The goal of the two equations and of the parameters used is to first make bigger, more random changes, and therefore, explore the whole solution space by simulating interactions with random other solutions and to make smaller, more targeted changes in the end when the changing degree is lower and when each object interacts with the best solution found so far [13].

## 4. Experiments

To answer the research questions, the AOA was implemented to solve different cases of the IRP. The implementation was conducted by creating a C# Windows Forms application in Visual Studio 2022.

### *4.1. Representation of solutions*

The solution is represented by the priority values and quantity values as vector components. Priority values, which are initialized as random numbers between 0 and 100, are used to determine the sequence in which customers are served by a depot in a specific period. The higher the priority compared to other customers, the earlier the customer is served. If all the quantity a depot produces

has been assigned to customers, then the remaining customers with lower priorities are not served. A priority value is also used to define the sequence of different trucks in the extended case. Trucks with higher priorities are planned first, and therefore, deliver to the customers with higher priorities.

Quantity values define how much of a product a customer receives from a specific depot during a specific period. This quantity is only delivered if the depot still has enough left to deliver it. Otherwise, if, for example, the customer has a lower priority, then the quantity of the customer is reduced, or the customer is not served at all.

These values allow us to apply the AOA to the considered optimization problem and to deal with the various constraints. Table 1 shows the pseudocode for the AOA adapted to the IRP. After generating an initial set of solutions, the algorithm iterates through its main loop for a prespecified number of times. Inside this loop, there is another one that considers each of the current solutions. For each solution, an interacting solution is determined (see above). Then, a further embedded loop runs through all planning periods. Volume, density, acceleration, transfer factor, changing degree, and F-Variable are calculated according to the general AOA. The updates rules are based on the specifications in Section 3.3. Then, the currently considered solution is updated (i.e., priorities and quantities). Based on this, the corresponding delivery requirements are calculated and a procedure to build a complete solution for the current period is carried out which first distributes the available stock to satisfy the customer demand required in the current period in the sequence of decreasing priorities. Then, if the depot can deliver the additional quantity, then the rest of the customers are processed and sorted by priority. Additionally, this procedure determines the specific quantities being shipped by one or several trucks and the respective tours.

**Table 1.** Pseudocode for the AOA Implementation adapted to the IRP (basic case).

```
Generation of initial solutions:
    for (int iteration = 0; iteration < maxIter; iteration++) {
        for each solution in solutions {
            interacting_Object = find_Interacting_Object(solution);
            for each timePeriod in timePeriods {
                    for each AOA_Basic_Solution_Space in Sol_Spaces {
                            update_Volume(bestSolution);
                            update_Density(bestSolution);
                            update_Acceleration(interactingObject);
                            update_TransferFactor();
                            update_ChangingDegree();
                            update_FVariable();
                            normalize_Acceleration();
                            update(best_Solution, interacting_Object);
                    }
                    define_Delivery_Requirements();
                    build_Solution(timePeriod);
            }
        }
        best_Solution = find_Best_Solution(solutions).Clone();
    }
```

### *4.2. Parameters of the optimization methods*

#### 4.2.1. AOA parameters

In the solution update formulas of the AOA, there are two parameters $x$ and $y$, which can be adapted to the specific use case. Test case TC-03 was used to identify the optimal values for $x$ and $y$ based on different settings of $x$ for 2 to 8 and for $y$ from 2 to 6. The population size and number of iterations were set to 100. The outlined costs correspond to the average result after five runs with the same settings. The results show that the parameter values proposed in [13] ($x = 2$, $y = 6$) do not work well on the IRP. $x$ needs to be larger than $y$ to obtain a satisfactory result. The best optimization was achieved with $x = 6$ and $y = 2$. These values were used in further tests of the proposed algorithm.

#### 4.2.2. Quantity parameters

A specific parameter implemented for the inventory routing scenario is the quantity threshold $q$. This parameter attempts to reduce the probability that a customer is served by multiple trucks in the same time period if it is not necessary. If decreasing or increasing the quantity by less than $q$ multiplied by the quantity determined by the algorithm would prevent the current customer from being served by multiple trucks, the quantity is reduced or increased accordingly. Tests varying this parameter from 0.1 to 0.5 demonstrated that the best parameter for $q$ was $q = 0.5$.

#### 4.2.3. Parameters for the GA and ACO

For the number of iterations and the population size in GA and ACO, values as specified in Section 4.5 are chosen to allow for a fair comparison. The GA uses further specific parameters for crossover and mutation, which were chosen as suggested in [5]: The GA uses a single-point crossover with probability 30% and a mutation probability of 20% (for a swap of customer priorities or customer quantities in a specific period). Regarding specific parameters for ACO, for alpha and beta, the values alpha = 4 and beta = 4 were applied based on initial testing of the methods. Other ACO parameters were chosen according to [31].

### *4.3. Scenarios*

Table 2 describes the IRP scenarios considered in this study. The basic and extended cases were used to confirm that the proposed AOA can be applied to both basic and extended cases of IRPs. In the ACO and GA comparison cases, the solution resulting from the AOA was compared to the solution found by the respective algorithms. For the presented scenarios, we randomly generated suitable test instances as suitable standard test instances for the considered problem types are not publicly available. For that, we considered plausible values for customer locations, demand, initial stock, holding costs, depot locations, production, and vehicle capacities, as further described in Table 3.

**Table 2.** Experiments - Scenario description.

| Scenario | Basic Case | Extended Case | ACO Comparison | GA Comparison |
|---|---|---|---|---|
| Time period | Discrete time periods | Discrete time periods | Discrete time periods | Discrete time periods |
| Customer | Customers have an initial stock of goods and constant consumption. | Customers have initial stock, a maximum stock, and variable consumption. | Customers have an initial stock of goods and constant consumption. | Customers have an initial stock of goods and constant consumption. |
| Depot | One depot | Multiple depots | One depot | Multiple depots |
| Product | One product | Multiple products | Multiple products | One product |
| Truck | All trucks have the same capacity. | Trucks have variable capacity. | All trucks have the same capacity. | All trucks have the same capacity. |

**Table 3.** Scenario generation logic.

| Object | Data Element | Logic |
|---|---|---|
| Customer | Location | The location data consisting of two variables to define the position of the customer on a plane was taken from an existing data set. Different customers have different locations. |
| Customer | Consumption | Randomly generated variable with a uniform distribution between 10 and 100. The random generation was executed for each customer individually so each customer can have a different consumption. |
| Customer | Initial Stock | Randomly generated variable with a uniform distribution between 0 and 200. The initial stock cannot be greater than the total consumption of the customer. Therefore, if total consumption was less than 200, the maximum initial stock was set to be smaller than this total consumption. |
| Customer | Holding Costs | Randomly generated variable with a uniform distribution between 0.1 and 0.5. The random generation was executed for each customer individually so each customer can have a varying holding cost. |
| Depot | Location | The location data consisting of two variables to define the position of the depot on a plane was taken from an existing data set. There is only one depot in the basic scenario so there is only one depot location. |
| Depot | Production | The production of the depot was set to the sum of the consumptions of all customers. |
| Truck | Capacity | Randomly generated variable divisible by 100 with a uniform distribution between 100 and 1000. This allows the generation of scenarios where one truck might not be enough to serve a customer but also scenarios where a truck can deliver to multiple customers. As there is only one truck type in the basic scenario this variable is only determined once and is valid for each truck that is used in the scenario. |
| Time Period | Time Periods | Randomly generated variable with a uniform distribution between 10 and 100. This allows testing longer and shorter scenarios. |

*4.4. Test criteria*

The following five metrics are considered for each test run to assess whether the AOA can be applied to a basic scenario of the IRP. 1) The Found Initial Solution (Found IS) evaluates whether the algorithm can produce initial solutions for the selected population size; the optimization algorithm can be applied only if these initial solutions are found. 2) The criterion Cost Best Initial Solution (Cost IS) analyzes the cost of the best initial solution. This measure is important as a comparison metric to investigate whether the algorithm improves on the initial solutions. This cost is the sum of the routing and inventory holding costs of the best initial solution produced by the program. 3) The criterion Cost Best Final Solution (Cost FS) represents the cost of the best solution that the algorithm has found after applying the algorithm to the selected number of iterations. This cost is again the sum of the routing and inventory holding costs. 4) The criterion Improvement (Imp %) is the percentage that the algorithm improves the best initial solution. This percentage is nonnegative as the algorithm returns the best initial solution if no better solution is found. Therefore, if the algorithm fails to produce a better solution, then the percentage is 0%. 5) Finally, the Algorithm Runtime (Algo RT) measures the runtime of the algorithm including initial solution generation and execution of the specified number of iterations. By collecting this metric for the AOA and the comparison algorithms (GA and ACO), a comparison can be made to identify the faster algorithm. The runtime is measured in milliseconds.

Based on these metrics, the following judgment is made to determine whether the algorithm can improve the initial solution: If an initial solution was found and the improvement was greater than 0%, then the algorithm was effective for the given scenario.

To compare the AOA with the GA and ACO (with parameter settings as described in Section 4.2), the following evaluations were considered: 1) The Final Cost Comparison (Comp Cost) considers the difference between the cost of the best-found final solutions produced by the algorithms. As the basis, the costs of the comparison algorithm (GA or ACO) are taken. Therefore, if the value is negative, then the AOA returns a better solution; if the value is positive, then the GA can find a better solution. 2) The Cost Result (Result Cost) indicates the algorithm that achieves better results in the cost comparison of a scenario in terms of achieving better final solutions. 3) The Runtime Comparison (Comp RT) considers the percentage by which the runtime of the AOA deviates from the runtime of the comparison algorithm. If the number is negative, then the AOA is executed more quickly, whereas a positive number indicates the comparison algorithm as the faster algorithm. 4) The Runtime Result (Result RT) indicates the algorithm with the better runtime when executing a scenario.

To obtain comparable runtime results, the following measures were taken: 1) While the tests were running, the computer was not used for other irrelevant activities. 2) Prior to the start of the tests, the CPU and RAM resources were monitored to ensure that no program with high resource usage was running. 3) After the start of the tests, the CPU and RAM resources were monitored again to ensure that sufficient capacity was left for the computer to handle background processes, such as updates.

Each test was run five times. To compare the two algorithms, the average runtimes of these five runs were compared. This further reduces the impact of background processes and other factors on the results. The tests were performed on a Windows 10 desktop PC with an Intel i7-5820k CPU and 16 GB RAM.

### *4.5. Test cases*

Table 4 presents the test cases used in our study. Each scenario was tested with three different settings regarding the population size and the number of iterations with parameters according to Table 5. Truck capacity is a unitless measure referring to the number of items that can be simultaneously transported.

**Table 4.** Overview of basic test cases.

| Test Case | Scenario | No. of Customers | Time Periods | Depots | Products | Truck Capacity |
|---|---|---|---|---|---|---|
| TC-01 | Basic | 3 | 61 | 1 | 1 | 100 |
| TC-02 | Basic | 31 | 33 | 1 | 1 | 900 |
| TC-03 | Basic | 50 | 73 | 1 | 1 | 500 |
| TC-04 | Basic | 59 | 25 | 1 | 1 | 200 |
| TC-05 | Basic | 200 | 40 | 1 | 1 | 1000 |
| TC-06 | Extended | 6 | 90 | 2 | 4 | Variable |
| TC-07 | Extended | 31 | 39 | 2 | 4 | Variable |
| TC-08 | Extended | 50 | 67 | 3 | 3 | Variable |
| TC-09 | Extended | 59 | 30 | 3 | 2 | Variable |
| TC-10 | Extended | 200 | 33 | 2 | 3 | Variable |
| TC-11 | GA Comp. | 6 | 85 | 2 | 1 | 100 |
| TC-12 | GA Comp. | 31 | 29 | 2 | 1 | 500 |
| TC-13 | GA Comp. | 50 | 44 | 3 | 1 | 200 |
| TC-14 | GA Comp. | 59 | 37 | 3 | 1 | 700 |
| TC-15 | GA Comp. | 200 | 25 | 2 | 1 | 1000 |
| TC-16 | ACO Comp. | 3 | 100 | 1 | 2 | 100 |
| TC-17 | ACO Comp. | 31 | 100 | 1 | 3 | 200 |
| TC-18 | ACO Comp. | 50 | 28 | 1 | 4 | 800 |
| TC-19 | ACO Comp. | 59 | 44 | 1 | 2 | 500 |
| TC-20 | ACO Comp. | 200 | 30 | 1 | 4 | 1000 |

**Table 5.** Test case parameters for the basic case.

| Test Cases | AOA population size | AOA no. of iterations |
|---|---|---|
| TC-XY-01 | 100 | for comparison with GA: 110in other cases: 100 |
| TC-XY-02 | 100 | for comparison with GA: 1100 in other cases: 1000 |
| TC-XY-03 | for comparison with GA: 1100 in other cases: 1000 | 100 |

To facilitate a fair comparison, the parameters of the compared algorithms were adjusted to ensure that both algorithms evaluate the same number of solutions. For TC-20, only test TC-20-01 was performed because the run time of ACO was too long for the other two settings.

### *4.6. Test results for basic and extended cases*

Table 6 presents the results of the tests for the basic and extended cases described in Section 3.5. Each test was executed five times. The table shows the average values of these 5 test runs.

**Table 6.** Test results for basic and extended cases.

| Run No. | Found IS | Cost IS | Cost FS | Imp (%) | Result |
|---|---|---|---|---|---|
| TC-01-01 | True | 10844 | 9513 | 12.27% | Passed |
| TC-01-02 | True | 10752 | 9559 | 11.10% | Passed |
| TC-01-03 | True | 10352 | 9290 | 10.26% | Passed |
| TC-02-01 | True | 62615 | 53022 | 15.32% | Passed |
| TC-02-02 | True | 62392 | 52190 | 16.35% | Passed |
| TC-02-03 | True | 61414 | 52752 | 14.10% | Passed |
| TC-03-01 | True | 245,873 | 219,991 | 10.53% | Passed |
| TC-03-02 | True | 246,539 | 2179,99 | 11.58% | Passed |
| TC-03-03 | True | 243,843 | 218,461 | 10.41% | Passed |
| TC-04-01 | True | 74510 | 68873 | 7.57% | Passed |
| TC-04-02 | True | 74242 | 68325 | 7.97% | Passed |
| TC-04-03 | True | 73731 | 68182 | 7.53% | Passed |
| TC-05-01 | True | 499,769 | 450,189 | 9.92% | Passed |
| TC-05-02 | True | 501,054 | 448,043 | 10.58% | Passed |
| TC-05-03 | True | 498,854 | 447,423 | 10.31% | Passed |
| TC-06-01 | True | 40057 | 38683 | 3.43% | Passed |
| TC-06-02 | True | 40083 | 38736 | 3.36% | Passed |
| TC-06-03 | True | 39801 | 38763 | 2.61% | Passed |
| TC-07-01 | True | 182,872 | 172,651 | 5.59% | Passed |
| TC-07-02 | True | 182,618 | 170,780 | 6.48% | Passed |
| TC-07-03 | True | 181,464 | 170,784 | 5.89% | Passed |
| TC-08-01 | True | 230,462 | 212,272 | 7.89% | Passed |
| TC-08-02 | True | 229,529 | 212,542 | 7.40% | Passed |
| TC-08-03 | True | 228,069 | 211605 | 7.22% | Passed |
| TC-09-01 | True | 132,060 | 120,415 | 8.82% | Passed |
| TC-09-02 | True | 131,269 | 119,178 | 9.21% | Passed |
| TC-09-03 | True | 130,593 | 118,716 | 9.09% | Passed |
| TC-10-01 | True | 512,505 | 488,468 | 4.69% | Passed |
| TC-10-02 | True | 512,065 | 484,793 | 5.33% | Passed |
| TC-10-03 | True | 509,914 | 483,553 | 5.17% | Passed |

### *4.7. Test results of the algorithm comparison*

Table 7 presents the results of the test comparison cases described in Section 3.5. Each test was executed five times. The table shows the average values of these 5 test runs. Regarding the cost comparison, AOA performed best in 21 of 28 cases, whereas it achieved best results regarding runtime only in 10 cases.

**Table 7.** Algorithm comparison and test results.

| Run Number | Comp Cost | Result Cost | Comp RT | Result RT |
|---|---|---|---|---|
| TC-11-01 | –2.10% | AOA | +97.14% | GA |
| TC-11-02 | +0.94% | GA | +96.36% | GA |
| TC-11-03 | –2.84% | AOA | +85.64% | GA |
| TC-12-01 | –3.04% | AOA | +219.93% | GA |
| TC-12-02 | +1.67% | GA | +240.14% | GA |
| TC-12-03 | –3.81% | AOA | +234.23 | GA |
| TC-13-01 | –6.97% | AOA | +211.81% | GA |
| TC-13-02 | –2.37% | AOA | +255.72% | GA |
| TC-13-03 | –6.95% | AOA | +230.42% | GA |
| TC-14-01 | –2.86% | AOA | +257.72% | GA |
| TC-14-02 | –0.24% | AOA | +290.94% | GA |
| TC-13-03 | –3.86% | AOA | +283.60% | GA |
| TC-15-01 | –9.61% | AOA | +79.82% | GA |
| TC-15-02 | –6.31% | AOA | +101.46% | GA |
| TC-15-03 | –10.24% | AOA | +95.39% | GA |
| TC-16-01 | –5.04% | AOA | +20.88% | ACO |
| TC-16-02 | –6.08% | AOA | +20.15% | ACO |
| TC-16-03 | –5.63% | AOA | +11.68% | ACO |
| TC-17-01 | +0.42% | ACO | –72.32% | AOA |
| TC-17-02 | –0.34% | AOA | –73.86% | AOA |
| TC-17-03 | –0.15% | AOA | –74.93% | AOA |
| TC-18-01 | +7.01% | ACO | –84.64% | AOA |
| TC-18-02 | +6.37% | ACO | –84.95% | AOA |
| TC-18-03 | +7.41% | ACO | –86.29% | AOA |
| TC-19-01 | –12.56% | AOA | –89.71% | AOA |
| TC-19-02 | –12.68% | AOA | –90.53% | AOA |
| TC-19-03 | –12.56% | AOA | –90.76% | AOA |
| TC-20-01 | +21.35% | ACO | –98.51% | AOA |

### *4.8. Discussion*

The results show that the AOA can be used to improve solutions for the IRP in both basic and extended scenarios. Better solutions than the initial solutions are found in all scenarios. In the basic scenario, the improvements are greater than in an extended scenario where the larger number of constraints makes it more difficult to find better solutions.

Based on these results, the first sub-research question can be answered. By implementing solution components based on priority and quantity, the AOA can be applied to the IRP. However, to allow the algorithm to converge to better solutions, certain adaptations are necessary. The variable q increases the chance that there are no delivery splits between customers, which reduces the cost of the solution returned by the algorithm.

In the conducted experiments, the GA converges much slower than the AOA. This is because changes from parent solutions to offspring solutions are much smaller compared to the progress in the AOA. While the AOA can adapt the priority and quantity of all customers for all depots and time periods, which can completely change the resulting solutions, the GA uses a different approach. An adapted solution is either a simple combination of two parent solution or the priorities and quantities of only one or a small number of customers are adapted. This limits the area in which the algorithm can search, especially when only a small number of iterations are used. The higher the number of iterations, the better the GA performance. A reason for this could be that smaller changes help the algorithm perform a good local search in the neighborhood of the current solution, which results in an even better solution. In the larger scenarios TC-13, TC-14, and TC-15, this did not result in a better solution than the solution found by the AOA. These scenarios are larger than scenarios TC-11 and TC-12, and the changes made by the GA were not sufficiently large for a full global search. In TC-11, increasing the number of iterations helped the GA find a better solution than the AOA while also increasing the number of iterations in this algorithm. However, the best solution for this scenario was found by the AOA when the population size was increased instead. The exception is TC-12. In this scenario, increasing the number of iterations helped the GA find the best solution.

The AOA obtained the best solution in 4 of 5 scenarios. Therefore, we conclude that the proposed algorithm can find more cost-effective solutions to the IRP than the GA. Therefore, regarding the third research question, we can confirm that the AOA can find more cost-effective solutions to IRPs than the GA in a majority of considered test cases.

In comparison, the GA had a much faster runtime compared to the AOA. The reason for this can again be attributed to its approach to solutions adaptation. In the simplest case, a solution adaptation in the GA performs a local search by just swapping two customers, including their quantity. In comparison, the AOA must adapt every component of a solution vector variable, which increases the computing resources required and the runtime. Thus, regarding the second sub research question, we conclude that the statement that the AOA has a smaller runtime than the GA must be rejected as the GA proved to be faster in the conducted experiments.

Considering the quality of the final solutions when comparing ACO and the AOA, the results of the tests are mixed. In some tests the AOA performs better, while ACO provides better results in other cases. However, regarding the runtime, there is a clear distinction. Scenario TC-16, which is very small, can be handled faster by ACO. However, as soon as scenarios increase in size, ACO becomes very slow. For example, for TC-18, the AOA was more than five times faster. The reason for this difference is as follows: To process customers in the AOA, the program just iterates through a list of customers from top priority to the lowest priority. On the other hand, ACO iterates through all customers after each truck stop to find the next best customer based on the distance and pheromone concentration. The runtime rapidly increases when more customers are added to the scenario. A single run of TC-20-01 with 200 customers takes more than 6 hours to complete. Therefore, when comparing the runtime of adapting a solution, the AOA is the clear winner. Hence, the part of the second sub-research question,

where the runtimes of generating and evaluating an adapted solution with the AOA and with ACO are compared, can be answered. The statement that the AOA runs faster than ACO can be confirmed.

However, slow processing in ACO has a clear advantage: The algorithm has better knowledge to find a good sequence in which customers should be served. In addition to the pheromone concentration, the algorithm can make use of the distance between the current node and other nodes to select a closer customer node for the next stop. Therefore, in two of five scenarios (TC-18 and TC-20), ACO performed significantly better than the AOA. In these scenarios, the AOA cannot find a better solution than ACO when increasing the runtime, either by increasing the population size or the number of iterations. The solution remains worse than the ACO solution even after increasing the number of iterations and population size parameters to a value where the runtime exceeds the runtime of ACO in which a better result is achieved.

According to the results, ACO created better final solutions in two of the five scenarios, and the AOA created better solutions in three of the five scenarios. Thus, each algorithm has specific scenarios in which the performance is better than that of the other algorithm. For this reason, the third sub-research question regarding the most cost-effective solution method can be answered by the statement that it depends on the scenario. The AOA performs better in small scenarios and in scenarios where quantities and holding costs are more important. ACO does not consider holding costs very well; instead, it only takes the cost of the route into account when choosing the next customer. Therefore, ACO performs better in cases where determining the transport routes is more crucial (e.g., in scenarios with many customers).

Our results confirm existing results regarding the suitability of ACO [31] and GAs [5] regarding their suitability for IRPs, although different formulations of the IRP were used in our study. Regarding the comparative results achieved in our study, there is a lack of similar studies in the literature, not only regarding AOA. We could not find any study comparing ACO and a GA for an IRP, even though both approaches are very established and popular for such types of problems.

## 5. Conclusions and managerial insights

Solving the IRP is an important challenge for suppliers that need to distribute goods to customers under a vendor-managed inventory agreement or other conditions that allow them to plan inventory levels at customers. In our study, we considered the AOA to solve the IRP, which is rather new and has not yet been applied to this problem. By representing the problem with priority variables, which define the sequence in which customers are served, and with quantity variables, which define how much quantity is shipped to the customer, the algorithm was able to successfully improve solutions in all tested scenarios during its execution. While more difficult scenarios increase the number of dimensions and thus the runtime, the algorithm still manages to find a solution.

We compared the AOA with two other metaheuristics, a GA and ACO. The proposed algorithm has a faster runtime than ACO; however, it is slower than the GA. Regarding the cost of the final solution, the GA performs best in small scenarios and returns better solutions. The larger the scenarios, the smaller the changes made by the GA, and the slower the algorithm progresses. Therefore, the AOA finds a better solution in four out of five scenarios. ACO returns better solutions in scenarios where the transport costs are more important. The algorithm performs well in finding optimal paths for trucks but cannot handle stock levels and holding costs very well. If the customer nodes close to a depot have very high holding costs, then ACO still delivers high quantities to these nodes, which can significantly

increase the costs of generated solutions. In some cases, this leads to situations in which ACO performs very poorly and struggles to find better solutions than the initial solution. On the other hand, the AOA exhibits a poorer route optimization performance. As a result of these effects, the AOA finds the best overall solution in three out of five scenarios in comparison to ACO.

Based on our findings, we can conclude that all the analyzed optimization algorithms have different advantages. Although the AOA does not find the best solution in all scenarios, the algorithm is very adaptable and can converge in a variety of scenarios. Although not having an exceptionally good runtime compared with the GA, the AOA is still comparably fast, as ACO is much slower in most cases. This knowledge can help organizations that must implement an algorithm to address the IRP by evaluating potential solutions. As each organization has different requirements, the ant colony algorithm may be the best choice for some organizations, whereas other organizations may prefer the speed and versatility of the AOA. However, as many more algorithms have been applied to the IRP than the three algorithms tested in our study, more research is necessary to better understand the exact strengths and weaknesses, and therefore, the benchmarking of the AOA compared to existing solutions.

However, there are various limitations in our study: First, the range of IRP types is limited and does not reflect the full range of situations found in practice; next, the analyzed test instances do not fully reflect the variety in business settings; and finally, we only compared three metaheuristics from a large number of such methods which have been developed and, so far, mostly been investigated for other problem types.

To address the mentioned limitations, a wider range of problem types could be explored, including further possibilities to adapt metaheuristics to make them suitable to such problem types. There are a huge number of such possibilities which should stimulate future research. Additionally, more settings regarding the size and number of test instances should be investigated to improve the benchmarking results. Future studies should enhance the experimental settings accordingly in order to get deeper insights into the performance of the compared algorithms.

Moreover, future research could investigate further adaptations to the algorithm to improve its ability to generate better solutions to the IRP. An algorithm that combines the advantages of the GA and ACO with those of the AOA could potentially help generate even better solutions. The advantage of the GA is that it can perform better local searches. When generating a new solution, not all customer sequences and not all quantities are adapted. Thus, the algorithm can better evaluate the impact on small changes resulting from a useful local search. The AOA could do the same by, for example, only running the adaptation formula on a randomly chosen part of the solution space (e.g., only adapting the quantities for one specific customer) in later time periods. Parts of ACO could also be introduced. Instead of defining the routes using only the priorities returned by the AOA, the routes could be determined according to the logic of both algorithms. The priority determined by the AOA can be included in the formula used by ACO to determine the next node. However, quantities and truck sequences could still be determined by only using the AOA results. These changes could help address the problem that ACO drops off large quantities on customers with high holding costs while still having the advantage of the very efficient routes that ACO can create.

In addition, we suggest comparing the AOA for the considered types of problems with further metaheuristics. Based on our findings, we assume that these will also show various strengths and weaknesses such as for the GA and ACO, which could stimulate the further development of methods, especially hybrid metaheuristics. Regarding advanced methods, one certainly promising direction is to

further explore machine learning, especially reinforcement learning, as successfully demonstrated already for other type of optimization problems [32].

However, an even more important aspect from a practical point of view is the general availability of software for supporting IRP related decisions: As far as we know, there is currently no software in widespread use to support companies in their logistics planning with solving the related IRPs. If we consider the vehicle routing problem as the most important subproblem, then the situation was similar 20 years ago. Today, however, there are many software tools available which support transport service providers with the optimized planning of routes. We assume that the situation with the IRP will evolve in a similar way. For the introduction and usage of such software, not too much is needed in addition to vehicle routing scenarios. The main requirement is transparency of inventory levels among the vendors and customers and the agreement on centralized planning as in the above-mentioned vendor managed inventory scenarios. However, the planning could be done either on the customer side, on the vendor side, by an in-volved transport service provider, or by an independent service provider such as a software company. We assume that large software providers in the area of Advanced Planning and Scheduling software will take the step of providing software solutions to the IRP in their commercial products offered to companies.

Using real-time data (such as from current sales and forecasts or traffic conditions), updated plans for shipment quantities and tour details can be calculated frequently. A mix of methods such as the AOA, GA, and ACO might be useful in a respective information system to allow for a flexible method choice based on current requirements regarding the problem size and the computation time.

## Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

## Conflict of interest

The authors declare there is no conflict of interest.

## References

1. L. Bertazzi, M. G. Speranza, Inventory routing problems: An introduction, *EURO J. Transp. Logist.*, **1** (2012), 307–326. https://doi.org/10.1007/s13676-012-0016-7
2. C. Archetti, M. G. Speranza, The inventory routing problem: The value of integration, *Int. Trans. Oper. Res.*, **23** (2016), 393–407. https://doi.org/10.1111/itor.12226
3. H. Shaabani, A literature review of the perishable inventory routing problem, *Asian J. Shipping Logist.*, **38** (2022), 143–161. https://doi.org/10.1016/j.ajsl.2022.05.002
4. I. T. Pere, H. M. Repolh, R. Martinelli, N. J. Monteiro, Optimization in inventory-routing problem with planned transshipment: A case study in the retail industry, *Int. J. Prod. Econ.*, **193** (2017), 748–756. https://doi.org/10.1016/j.ijpe.2017.09.002
5. A. Hiassat, A. Diabat, I. Rahwan, A genetic algorithm approach for location-inventory-routing problem with perishable products, *J. Manuf. Syst.*, **42** (2017), 93–103. https://doi.org/10.1016/j.jmsy.2016.10.004

6. F. A. Touzout, A. Ladier, K. Hadj-Hamou, An assign-and-route matheuristic for the time-dependent inventory routing problem, *Eur. J. Oper. Res.*, **300** (2022), 1081–1097. https://doi.org/10.1016/j.ejor.2021.09.025
7. E. Nikzad, M. Bashiri, F. Oliveira, Two-stage stochastic programming approach for the medical drug inventory routing problem under uncertainty, *Comput. Ind. Eng.*, **128** (2019), 358–370. https://doi.org/10.1016/j.cie.2018.12.055
8. K. T. Malladi, T. Sowlati, Sustainability aspects in inventory routing problem: A review of new trends in the literature, *J. Cleaner Prod.*, **197** (2018), 804–814. https://doi.org/10.1016/j.jclepro.2018.06.224
9. C. Archetti, L. Bertazzi, Recent challenges in routing and inventory routing: E-commerce and last-mile delivery, *Networks*, **77** (2021), 255–268. https://doi.org/10.1002/net.21995
10. A. C. Baller, M. van Ee, M. Hoogeboom, L. Stougie, Complexity of inventory routing problems when routing is easy, *Networks*, **75** (2020), 113–123. https://doi.org/10.1002/net.21908
11. M. Çelik, C. Archetti, H. Süral, Inventory routing in a warehouse: The storage replenishment routing problem, *Eur. J. Oper. Res.*, **301** (2022), 1117–1132. https://doi.org/10.1016/j.ejor.2021.11.056
12. N. I. Saragih, S. N. Bahagia, Suprayogi, I. Syabri, A heuristic method for location-inventory-routing problem in a three-echelon supply chain system, *Comput. Ind. Eng.*, **127** (2019), 875–886. https://doi.org/10.1016/j.cie.2018.11.026
13. F. A. Hashim, K. Hussain, E. H. Houssein, M. S. Mabrouk, W. Al-Atabany, Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems, *Appl. Intell.*, **51** (2021), 1531–1551. https://doi.org/10.1007/s10489-020-01893-z
14. K. G. Dhal, S. Ray, R. Rai, A. Das, Archimedes optimizer: Theory, analysis, improvements, and applications, *Arch. Comput. Methods Eng.*, **30** (2023), 2543–2578. https://doi.org/10.1007/s11831-022-09876-8
15. C. Archetti, I. Ljubić, Comparison of formulations for the inventory routing problem, *Eur. J. Oper. Res.*, **303** (2022), 997–1008. https://doi.org/10.1016/j.ejor.2021.12.051
16. Z. Cui, D. Z. Long, J. Qi, L. Zhang, The inventory routing problem under uncertainty, *Oper. Res.*, **71** (2022), 378–395. https://doi.org/10.1287/opre.2022.2407
17. A. Gutierrez-Alcoba, R. Rossi, B. Martin-Barragan, T. Embley, The stochastic inventory routing problem on electric roads, *Eur. J. Oper. Res.*, **310** (2023), 156–167. https://doi.org/10.1016/j.ejor.2023.02.024
18. S. Prakash, I. Mukherjee, G. Soni, R. Piplani, An enhanced multiobjective inventory routing model to meet sustainable goals for assembly supply network under uncertainty, *Ann. Oper. Res.* **364** (2026), 1289–1339. https://doi.org/10.1007/s10479-024-06222-y
19. T. Liu, J. Liu, H. Xu, A multi-population multi-objective maritime inventory routing optimization algorithm with three-level dynamic encoding, *Sci. Rep.*, **15** (2025), 13242. https://doi.org/10.1038/s41598-025-86091-y
20. X. Shang, G. Zhang, B. Jia, M. Almanaseer, The healthcare supply location-inventory-routing problem: A robust approach, *Transp. Res. Part E: Logist. Transp. Rev.*, **158** (2022), 102588. https://doi.org/10.1016/j.tre.2021.102588

21. K. Govindan, F. Salehian, H. Kian, S. T. Hosseini, H. Mina, A location-inventory-routing problem to design a circular closed-loop supply chain network with carbon tax policy for achieving circular economy: An augmented epsilon-constraint approach, *Int. J. Prod. Econ*., **257** (2023), 108771. https://doi.org/10.1016/j.ijpe.2023.108771
22. L. Song, Z. Wu, An integrated approach for optimizing location-inventory and location-inventory-routing problem for perishable products, *Int. J. Transp. Sci. Technol.*, **12** (2023), 148–172. https://doi.org/10.1016/j.ijtst.2022.02.002
23. M. R. Pourhassan, M. R. K. Roshandeh, P. Ghasemi, M. S. S. Bathaee, A multi echelon location-routing-inventory model for a supply chain network: NSGA II and multi-objective whale optimization algorithm, *Int. J. Supply Oper. Manage.*, **12** (2025), 81–104. https://doi.org/10.22034/ijsom.2023.109996.2804
24. J. Cao, J. Gao, B. Li, X. Wang, The inventory routing problem: A review, in *20th COTA International Conference of Transportation Professionals*, (2020), 4488–4499. https://doi.org/10.1061/9780784482933.385
25. O. Solyalı, H. Süral, An effective matheuristic for the multivehicle inventory routing problem, *Transp. Sci.*, **56** (2022), 1044–1057. https://doi.org/10.1287/trsc.2021.1123
26. J. Skålnes, S. T. Vadseth, H. Andersson, M. Stålhane, A branch-and-cut embedded matheuristic for the inventory routing problem, *Comput. Oper. Res.*, **159** (2023), 106353. https://doi.org/10.1016/j.cor.2023.106353
27. D. Laganà, E. Malaguti, M. Monaci, R. Musmanno, P. Paronuzzi, An iterated local search matheuristic approach for the multi-vehicle inventory routing problem, *Comput. Oper. Res.*, **169** (2024), 106717. https://doi.org/10.1016/j.cor.2024.106717
28. S. Charaf, D. Taş, S. D. P. Flapper, T. Van Woensel, A matheuristic for the two-echelon inventory-routing problem, *Comput. Oper. Res.*, **171** (2024), 106778. https://doi.org/10.1016/j.cor.2024.106778
29. M. Krisnawati, A. C. Monita, A. Sofiana, Solving inventory routing problems with crow search algorithm, *J. Ind. Eng. Int.*, **21** (2025), 41–48.
30. Z. Dai, Y. Zhou, B. C. Giri, Two heuristic algorithms for location-inventory-routing models involving two warehouses within multi-echelon supply chain networks, *Sci. Rep.*, **15** (2025), 43964. https://doi.org/10.1038/s41598-025-27614-5
31. L. Wong, N. H. Moin, Ant colony optimization for split delivery inventory routing problem, *Malays. J. Comput. Sci.*, **30** (2017), 333–348. https://doi.org/10.22452/mjcs.vol30no4.5
32. A. Tian, J. Pan, H. Lv, Optimizing train scheduling in heavy-haul railways using diversified cooperative deep reinforcement learning, *Transp. Res. Rec.*, **2680** (2026), 286–312. https://doi.org/10.1177/03611981251364832





AIMS Press

©2026 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0)