Erasmus School of Economics
Master Thesis Econometrics and Management Science
Operations Research and Quantitative Logistics

January 2022

Solving the Supplier Selection problem utilizing Multi-objective Ant Colony Optimization[1]

Name student:	Annelot Bosman
Student number:	472467
Supervisor:	dr. O. Karabağ
Second accessor:	prof. dr. APM Wagelmans

Abstract

This research considers the *Supplier Selection* problem provided by Médicines sans Frontières. This is a multi-objective problem within a humanitarian supply chain, which has not been considered in existing literature. The main goal of the research is to develop a tool that makes the procurement process more insightful for the decision makers. Five *Multi-objective Ant Colony Optimization* (MOACO) algorithms are proposed to solve the problem. It is found that the *crowded Population-based ant colony optimization* (c-PACO) results in the best set of non-dominated solutions and has the fastest solver-time. With at-least 20 million euro of the procurement budget being influenced by this decision tool and an average and best case cost reduction of 9.99 % and 17.10% respectively, using the decision tool is expected to safe the company anywhere from 1.08 to 3.42 million per year.

[1]The views stated in this thesis are those of the author and not necessarily those of the supervisor, second accessor, Erasmus School of Economics or Erasmus University Rotterdam.

Contents

1 Introduction

This research considers a supplier selection problem provided by Médicines sans Frontières (MSF). MSF is an international, independent medical humanitarian organisation that provides medical assistance in conflict areas and when natural disasters or deadly disease outbreaks occur (Bates, 2005). This provision of medical care is organised through missions based in countries all over the world. Each of these missions coordinates different projects. For instance, a project in Russia is focused on controlling the spread of drug-resistant tuberculosis and treating patients infected with the disease. In Sudan for example, one project is supporting the people that have been displaced due to wars and another in the same country is focused on treating covid-19 patients.

To successfully execute a project, supplies are essential. For instance, a vaccination campaign needs vaccines. These supplies are ordered by the managers of that project. Such an order usually consists of multiple types of goods, which will be called *items* in this research. The orders from all missions are collected at the headquarters of the organisation. At the headquarter suppliers are sourced, pricing agreements are made and purchase orders are placed with suppliers. The purchase orders will be called *purchases* in this research. Even though agreements are made with suppliers, in practice important factors such as the agreed-upon lead-time are often exceeded due to uncertainty. Usually, multiple suppliers are available to provide the same item, making this a multi-item multi-supplier supply chain problem.

The procurement problem considered in this research consists of deciding which items are purchased from which supplier, which will be called the *item assignment* in this research. The item assignment is created per mission order. The item assignment needs to adhere to certain restrictions, for instance, some suppliers might have a minimum order value (MOV) or minimum order quantity (MOQ). Considering one mission order, many different item assignments might be feasible. A decision on which item assignment is optimal cannot be made unambiguously as multiple objectives are considered in this research, which are total cost minimization and lead-time optimization. Furthermore, some of the factors contributing to the objectives, such as delivery time and production time, are subject to uncertainty.

It can be expected that an item assignment that is optimal for one of the objectives is not the optimal assignment for the other objectives. For instance, if one wants especially high-quality items, these might take longer to make than low-quality items, thus leading to a higher lead time. At the same time producing a high-quality item will in general lead to higher production costs and thus higher total costs and vice versa. Which objective is most important for a certain mission is decided by the decision-maker. Pareto optimality exists when considering two solutions none of the two solutions dominates the other. A solution dominates another if two conditions hold (Mishra & Harit, 2010). Firstly, none of the cost-vectors of the first solution is worse than the cost-vectors of the other and secondly, at least one of the cost-vectors of the first solution is better than the corresponding cost-vector of the second solution.

Currently, the item assignments are made manually by the decision-makers. A feasible item assignment is constructed by employees that each uses their own judgement, which is usually based on the lowest per-piece price of an item, not considering other factors contributing to the cost, such as delivery costs, as this information is simply not available to the decision-makers. This method is expected to not lead to optimal item assignments, especially for the lead time optimization objective, as it is impossible to test all allocations by hand. This research is expected to not only reduce procurement costs but will also exclude guesswork from the procurement process. Furthermore, it will omit the tedious and time-consuming process and change item assignment to a simple selection.

The goal of this research is to provide a *decision tool* such that the decision-makers can select the item assignment with the objectives best suited for the corresponding mission. To provide this tool we state the multiple research questions of which the first is : *Which methods have been proposed in the existing literature to solve problems similar to the assignment problem at hand?* To answer this question an extensive literature review was performed. We found that Ant Colony Optimization has been proposed in recent literature to solve the supplier selection problem as well as different variations of the Assignment Problem. Therefore, we will implement an Ant Colony algorithm and extend it with multiple state-of-the-art improvements. The next question is: *How do the selected methods perform in the context of the assignment problem at hand?* This question is answered by using quantitative metrics to evaluate the proposed methods.

The last question is: *Does the proposed decision tool provide added value compared to the status quo?* We do this by assuming the only goal of the organisation is cost minimization. Even-though cost minimization is not the most important factor in all item assignments, we can assume this is what decision-makers strove for in past as they had no other quantitative data to rely on. We will compare the historic costs of project cost with the expected minimal cost provided by the decision tool over the same orders.

The contribution of this research is threefold. Firstly, humanitarian supply chains have not been investigated in the context of the supplier selection problem, which makes this research a valuable contribution to existing literature. Secondly, the multi-item multi-supplier supply chain problem has not been formulated as a Generalized Assignment problem in the existing literature. Lastly, this study will investigate the Multi-objective ant colony optimization using real-life data for the first time in the context of the supplier selection problem as previously only simulated data was used.

This research continues as follows: A literature review is provided in Section 2. Section 3 describes the problem in detail. The methodology used to solve the problem is provided in Section 4. In Section 5 the data and results are described. Section 6 contains the discussion of the research questions and the conclusion of the research.

2 Literature

This section provides an overview considering relevant literature. First, a short introduction regarding the humanitarian supply chain literature is given. Then, research regarding multi-objective optimization is discussed. Thereafter, a description of the history of research regarding the supplier selection problem is presented. Then, more background regarding the combinatorial problem named the Assignment Problem is discussed. Lastly, recent research combining the supplier selection problem and Genetic Algorithms is discussed.

2.1 The humanitarian supply chain

Successful supply chain management has become a topic of great relevance in emerging competitive markets since the 1990s (Lambert & Cooper, 2000). A supply chain consists of all individual stages and parties involved in fulfilling the demand for a certain commodity (Chopra & Meindl, 2007). The supply chain includes, amongst orders, produce sourcing, transportation, production and information. As Chopra & Meindl (2007) state, the goal of a supply chain is creating supply chain surplus, which is often defined by profit. This surplus can be created by the provision of high-quality products or short lead-times.

In this research, a humanitarian supply chain is considered. The humanitarian supply chain differs from the commercial supply chain in the sense that the goal of the humanitarian supply chain is to provide suitable aid to the beneficiaries (R. M. Tomasini & Van Wassenhove, 2009). In their book, R. Tomasini et al. (2009) describe that one of the major difficulties in managing the humanitarian supply chain is the ambiguity of the goal of the supply chain. In the case of a disaster, disaster relief operations need a quick response and thus prefer short lead-times. This type of operation are usually very costly, hard to prepare for and often dangerous. On the other hand, the item sourcing should be as efficient as possible to get the biggest return out of the donated money.

2.2 Multi-objective optimization

In the previous section, we described the ambiguity in the goals of the humanitarian supply chain. To deal with this ambiguity we regard our problem as a multi-objective problem, with the following objectives: cost minimization and lead-time optimization. Multi-objective optimization is suited for optimization problems where there are more than one, often conflicting, objectives (Deb, 2014). The problem considered in this research is an example of such a problem. A high-quality product will generally have a longer production time and cost more than a low-quality product.

There are four classical methods in solving multi-objective problems of which objective weighting is the most popular. In objective weighting, each of the objectives considered is given a weight corresponding to their relative importance (Coello et al., 2007). A drawback of this method is that the weights need to be assigned beforehand and one solution is presented. In this research the determination of which objective is important is made by a decision-maker, making objective weighting inapplicable.

As there will be a decision-maker deciding which objective is important for which mission, our goal is to provide a comprehensive decision tool. The item assignments incorporated in the decision tool will be a set of Pareto optimal assignments. Andersen et al. (1996) state that providing a large set of Pareto optimal solutions, in our case item assignments, is important to make a well-informed decision. In recent literature, Genetic Algorithms (GA) are proposed for providing such a Pareto optimal solution set. Examples are the Non-dominated Sorting Genetic Algorithm (NSGA-II) method proposed by Deb et al. (2002), this the most used method (Qu et al., 2018), the Multi-objective Ant Colony optimization of which there are many versions and general GA's. Genetic algorithms have the advantage of global searching ability, while Ant Colony Optimization has an advanced feedback mechanism. Global searching ability means that the entire solution space is searched.

Multi-objectivity is important in the context of the problem at hand. The main driving factor for traditional supply chains is profit maximization, which is explicitly not the case for the humanitarian supply chain. This fact makes the multi-objectivity of high relevance as no explicit quantitative goal of the supply chain can be stated a priori. Humanitarian supply chains have not been investigated in the context of the supplier selection problem, which makes this research a valuable contribution to existing literature.

2.3 Supplier selection

Supplier selection is an important part of the operational decision making in any company, as Ghodsypour & O'Brien (1998) claim that 70% percent of production costs stem from purchasing goods and services and R. M. Tomasini & Van Wassenhove (2009) claim that in humanitarian supply chains that even 80% of the total costs stem from procurement and operations. The complicating factor in supplier selection is the number of quantitative and qualitative criteria a supplier needs to adhere to. Many different criteria can be defined, yet Weber et al. (1991) find that the three most used are net price, product quality and the capability to meet delivery schedules. Two of these factors are considered in this research.

Supplier selection is one of the first stages in the supply chain and affects all consecutive stages afterwards. The entire supplier selection process consists of the formulation of criteria that the supplier need to adhere to; pre-qualification of suppliers; supplier selection; monitoring of selected suppliers (Sonmez, 2006). This research considers the latter two steps of the supplier selection only, as the criteria and pre-qualification are already performed. These two steps could never be performed by a heuristic in the considered supply chain, as this requires expert knowledge about pharmaceutical practice.

The supplier selection problem has received considerable attention in recent years. Where older literature mostly considers the selection of supplier criteria and assigning weights to these criteria, in recent years the focus has shifted towards actual supplier selection. A systematic literature review, performed by Chai et al. (2013), shows three main categories of methods used. Namely, multi-criteria decision-making methods, mathematical programming methods and artificial intelligence (AI) techniques. In their latest survey, Chai & Ngai (2020) see a rising interest in AI techniques. This is no surprise as intelligent optimization methods have been shown to increase the probability of finding good solutions for highly complex problems (Luan et al., 2019) and the supplier selection problem is known to be complex.

To the best of our knowledge, the supplier selection problem has not yet been considered in the context of the humanitarian supply chain. This means the context of the problem will be different from the research conducted before. As previously stated, the goal of the humanitarian supply chain is explicitly not to maximize profit, even though cost minimization is relevant, this is interesting given that in general, this has been the motivation behind research towards the supplier selection problem.

2.4 Assignment problem

The Assignment problem is one of the fundamental problems in combinatorial optimization. The Assignment problem is that of assigning n jobs to n persons where each person can take on each job for a different cost to minimize the total cost. The first algorithm to solve this problem, known as the Hungarian method, was developed by Kuhn (1955). This algorithm was proven to be solvable in polynomial time by Munkres (1957). There are many variations of the Assignment problem, such as the Asymmetric Assignment problem where the number of jobs is not equal to the number of workers. An overview of the most famous Assignment problems is given by Burkard et al. (2012).

The Generalized Assignment problem is that of assigning jobs to workers, where multiple jobs can be given to each worker. Krumke & Thielen (2013) propose a Generalized Assignment problem with minimum quantity constraints (GAP-MQ). The GAP-MQ has the restriction that when a worker is employed, they need to have a minimum number of jobs or none at all (in the research they use the analogy of bins and items, such that the bins need to be filled to a certain point at minimum). Krumke & Thielen (2013) show that the GAP-MQ problem is strongly NP-complete. The problem considered in this research is similar to the GAP-MQ problem as a part of the suppliers has a MOV or MOQ. This means that the problem at hand is strongly NP-complete and cannot be solved to optimality in polynomial time (unless P =NP).

Formulating the multi-item multi-supplier supply chain problem as a Generalized Assignment problem has, to the best of our knowledge, not been proposed in previously published literature. Proposing to solve it using a well-known linear programming (LP) problem and investigating the use of this is a new approach. How the problem at hand relates to existing literature will be described in detail in the following section.

2.5 Genetic algorithms and the supplier selection problem

For solving Assignment problems Rizk & Arnaout (2012) found that the Ant Colony Optimization (ACO) outperforms branch-and-bound and global solvers in terms of solution quality and computation time. Stützle & Dorigo (1999) found that for the Quadratic Assignment problem the ACO combined with a local search is among the best heuristics for solving the complex problem. Even though the paper by Stützle & Dorigo (1999) is over twenty years old, a recent survey conducted by Chai & Ngai (2020) shows that the use of AI techniques and GA's in the context of the supplier selection problem is currently rising. We will investigate some of the proposed, state of the art methods for problems similar to the problem investigated in this research.

Luan et al. (2019) investigate the use of a hybrid ACO and GA method in the context of a multi-item, multi-supplier, multi-objective supplier selection problem with a quantity discount. Their work is inspired by the use of this hybrid method in the context of partner selection problems in virtual enterprises. Luan et al. (2019) find that the ACO and GA hybrid method has a good time and optimality performance based on their simulated numerical experiments. The problem studied by Luan et al. (2019) is similar to the problem at hand, except for the problem at hand deals with MOV and MOQ instead of quantity discounts. They also do not incorporate any uncertainty regarding performance parameters in their research, which is the case in the problem at hand. Furthermore, instead of providing a Pareto optimal front, they use TOPSIS, which is an advanced weighing mechanism, to assign weights to different objectives.

Abdollahzadeh & Atashgar (2017) investigate the maintenance of a system with multiple objectives and uncertainty in suppliers based on cost, performance and supplier behaviour. This uncertainty is modelled using scenario representation. They propose a multi-objective stochastic optimization model to formulate this problem and provide a Pareto optimal set using a Multi-objective Ant Colony Optimization (MOACO) algorithm. To prevent the case that the cardinality of the Pareto optimal set becomes too large to make a good decision, Abdollahzadeh & Atashgar (2017) employ a k-means clustering algorithm and consecutively a data envelopment method (DEA) to identify the representative solutions from each cluster. They state that the proposed model has high capability.

The above-mentioned methods are theoretical studies using simulated data. To the best of our knowledge, this study will investigate the ACO and MOACO using real-life data for the first time. Thus, the relevance of this research is threefold. For the first time, the supplier selection problem will be investigated in a humanitarian supply chain context, formulated as a Generalized Assignment Problem and the supplier selection problem will be solved using an ACO algorithm with real-life data.

2.6 Ant Colony Optimization

In this research, the Ant Colony Optimization meta-heuristic will be applied to the item assignment problem to find a set of non-dominated feasible solutions. The ACO was first proposed by Dorigo & Gambardella (1997) to solve the travelling salesman problem and has since been applied to many combinatorial optimization problems, such as the shortest path problem. The ACO has gained popularity due to its ability to find high-quality solutions for highly complex problems with reasonable computation time. The original ACO has been transformed and many improvements and extensions have since been proposed. One of those is the Multi-objective Ant Colony Optimization (MOACO). García-Martínez et al. (2007) conclude from their taxonomy that MOACO's are more competitive in the context of a multi-criteria travelling salesman problem than other algorithms that are often proposed such as the NSGA-II.

3 Problem description

The problem considered in this research is a multi-objective multi-item multi-supplier supply chain optimization problem. A feasible item assignment needs to adhere to some conditions. Firstly, the items need to be bought in at least the quantity as stated on the mission order. Secondly, for all suppliers selected, the MOV or MOQ needs to be attained, if there is one, except for some specific suppliers that give a penalty for not obtaining the MOV or MOQ. Thirdly, the remaining shelf-life of the perishable items need to be sufficient. Lastly, some countries require specific documents for imported goods, if this is the case those documents need to be available as well. The problem at hand is a *Multi-Objective Asymmetric Generalized Assignment* (MOAGA) problem.

This MOAGA problem is a generalization of the well known *Assignment problem.* The Assignment problem considers a situation where a set of jobs needs to be assigned to a set of workers, with as objective the minimization of costs. Each job is assigned to exactly one worker and each worker is assigned to exactly one job, creating a perfect matching. The worker can be seen as a supplier and the job as a single unit of a certain item. In the problem at hand, however, one supplier can be assigned multiple items in one or more units. One item type can be sourced partially from one supplier and partially from another. Not all available suppliers need to be assigned an item to supply. Furthermore, suppliers might uphold a MOV or MOQ. The original Assignment problem has one objective, while the problem at hand has multiple objectives.

Once the order has come in, the decision makers need to make an item assignment. This supplier selection problem is part of a supply chain that can be found in Figure 1.

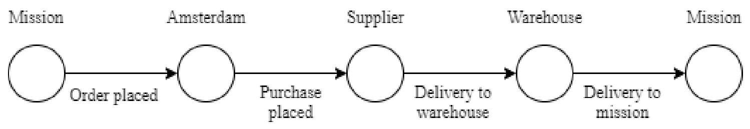


Figure 1: Graphic depiction of the supply chain of an item in the purchase policy.

The assignment problem considers two objectives. The first objective is the minimization of total costs. The total costs consist of per piece purchase price of an item, packaging costs, service costs and transportation costs. The second objective is the optimization of lead-time, this consists of the sum of the production and delivery time per item. This lead-time optimization minimizes the fraction of items that are expected to not be delivered within 28 days, we will refer to this as the *fraction of late lines.* The 28 days are chosen as this is a company policy.

3.1 Mathematical formulation of the item assignment problem

The MOAGA problem is defined as follows. Set I contains all the items that need to be assigned to suppliers within a certain mission order, falling under the purchase policy. Non-empty set V_i is defined as the set of suppliers available for item $i \in I$. The set of all possible suppliers that can at least supply one of the items in set I, is denoted by $\mathcal{V}$,

$$\mathcal{V} = \{(i, v) | v \in V_i, i \in I\}. \tag{1}$$

In this research two main objectives are considered, namely the minimization of the total cost of an item assignment, $f^1(x)$, and the minimization of the fraction of items having a delivery time longer than 28 days, $f^2(x)$. Here x is defined as a complete and feasible item assignment. The fact that the second objective considers 28 days is because of a company policy that they need to strive for a 28-day lead-time. There are other objectives, such as maximization of the remaining shelf-life of items and the quality of items, yet those will not be considered in the main research as they are secondary objectives.

The parameters influencing the objectives are subject to uncertainty. These parameters are the lead-time, defined as $t_{i,v}$, and the combined total of transportation and service costs charged by a supplier, defined as d_v. These uncertainties can be the result of disturbances occurring in the supply chain of the supplier or occurring during transportation. Examples of such disturbances are traffic, machine malfunctions or volatile fuel costs.

To handle these uncertainties, in the main part of this research, the objective functions will be made robust. To do this two new parameters are constructed for each vendor, namely one consisting of the average historical total lead-time for each possible item, defined as $L_{i,v}, \forall i \in I, v \in V_i$, and one consisting of average historical combined transport and service costs per order, defined as $D_v, \forall v \in \mathcal{V}$. These samples are used to estimate the average transportation cost and lead-time based on a certain item assignment x. This method is inspired by the work of Gutjahr (2004). They find that this method works well considering their simulated problem.

In this research multiple decision variables are considered. Variable $x_{i,v}$ is integer and denotes the number of items i assigned to supplier v, $\forall i \in I, v \in V_i$. Variable y_v is binary and equals 1 if supplier v is used in the item assignment and 0 otherwise, $\forall v \in \mathcal{V}$. Variable P_v is binary and equals 1 if the MOV is not attained and 0 otherwise, $\forall v \in \mathcal{V}$. The objective values can be found in Equation (2) and (3).

Objective (2) is to minimize the expected sum of the cost of purchasing each item, the expected penalty cost and the expected transportation and service costs, called AFAM costs. These AFAM costs are calculated relative to the purchase value of an item compared to the total purchase value assigned to that supplier. Objective (3) calculates the expected fraction of late-lines as the fraction of items that are expected to be delivered later than 28 days.

$$\min \mathbb{E}(f^1(x)) = \sum_{i \in I} \sum_{v \in V_i} c_{i,v} \cdot x_{i,v} + \sum_{v \in \mathcal{V}} pv_v \cdot P_v + \sum_{i \in I} \sum_{v \in V_i} D_v \frac{c_{i,v} x_{i,v}}{\sum_{j \in I} c_{j,v} x_{j,v}} \tag{2}$$

$$\min \mathbb{E}(f^2(x)) = \frac{1}{|I|} \sum_{v \in V_i} \sum_{w \in L_{i,v}} l_{i,w} \cdot x_{i,v} \tag{3}$$

where parameter $c_{i,v}$ is the per piece price of purchasing item i from supplier v. Parameter pv_v is the penalty cost of not attaining the minimum order value that supplier v upholds, if such a penalty exists. Parameter d

Parameter $l_{i,v}$ is a binary variable that attains the value of 1 if an item i is expected to not be delivered within 28 days when ordered from supplier v, thus when $L_{i,v}$ is bigger than 28, and 0 otherwise.

Constraint(4) ensures that at least the number of needed items are ordered. Parameter q_i is defined as the quantity ordered of item i. The number of items can be higher as suppliers often do not sell an item per piece, but per more at a time, constraint (5) ensures that the right quantity is ordered from each supplier such that it adheres to the purchase unit. Here, a dummy integer variable $\sigma_{i,v}$ is used that ensures that the fraction in constraint (5) is also an integer value. Parameter $u_{i,v}$ is defined as the purchase unit for an item i when purchased from supplier v.

$$\sum_{v \in V_i} x_{i,v} \geq q_i \qquad \forall i \in I \tag{4}$$

$$\frac{x_{i,v}}{u_{i,v}} = \sigma_{i,v} \qquad \forall i \in I, v \in V_i \tag{5}$$

Constraint (6) makes sure that when items are ordered from vendor v variable y_v attains the value 1. This is necessary for the transport and service cost calculation in the total cost objective function. To do this a large number M_1 is employed. M_1 needs to be bigger than the total quantity of items ordered from vendor v.

$$\sum_{i \in I | v \in V_i} x_{iv} \leq M_1 \cdot y_v \qquad \forall v \in \mathcal{V} \tag{6}$$

Define $V_{non} \subseteq \mathcal{V}$ as the set of suppliers that has a strict MOV, so it is not possible to get a penalty when the MOV is not met. Constraint (7) determines whether the MOV for supplier v is met, such that it influences variable P_v. Parameter M_2 is a sufficiently larger number. If supplier v is an element from set V_{non}, the MOV must be met and this is ensured via constraint (8). Constraint (9) ensures that the MOQ for supplier v is met if any items are purchases from that supplier.

$$MOV_v \cdot y_v - \sum_{i \in I | v \in V_i} c_{iv} \cdot x_{iv} \leq M_2 \cdot P_v \qquad \forall v \in \mathcal{V} \tag{7}$$

$$P_v = 0 \qquad \forall v \in V_{non} \tag{8}$$

$$MOQ_v \cdot y_v \leq \sum_{i \in I | v \in V_i} x_{iv} \qquad \forall v \in \mathcal{V} \tag{9}$$

Constraint (10) until (13) define the intervals of each of the variables used.

$$x_{i,v} \in \mathbb{N} \qquad \forall i \in I, v \in V_i \tag{10}$$

$$y_v \in \{0,1\} \qquad \forall v \in \mathcal{V} \tag{11}$$

$$P_v \in \{0,1\} \qquad \forall v \in \mathcal{V} \tag{12}$$

$$\sigma_{i,v} \in \mathbb{N} \qquad \forall i \in I, v \in V_i \tag{13}$$

4 Methodology

Ant Colony Optimization, first proposed by Dorigo & Gambardella (1997), is a meta-heuristic belonging to the concept of Swarm Intelligence. The ACO heuristic has gained popularity due to being able to attain (near) optimal solutions, its flexibility and robustness (Falcón-Cardona et al., 2020). The ACO meta-heuristic can be used to solve any problem that can be represented as a graph, which makes it broadly applicable. The meta-heuristic has recently gained interest in the field of multi-objective optimization and the results have been promising (Falcón-Cardona et al., 2020). ACO algorithms dealing with multiple objectives are called Multi-Objective Ant Colony Optimization (MOACO) algorithms.

The concept underlying the ACO meta-heuristic is that of the foraging behaviour of real-life ants. The goal of the ACO algorithm is to find the best solution to the problem at hand, employing artificial ants to find these solutions. Real-life ant colonies find the shortest path between their anthill and a food source by laying down pheromone trails on the path they are taking (Deneubourg & Goss, 1989). Ants prefer paths with stronger pheromone trails, making this a self-enforcing process. The ACO meta-heuristic employs artificial ants that are analogue to real-life ants in the sense that they can detect pheromones and deposit pheromones on the arcs of a graph representing the problem to be solved. The strength of the pheromone trails represents the quality of the solution the path of the ant suggests.

The main algorithm employed in this research is based on the work of Alaya et al. (2007), which will be called g-ACO (generic ACO). Their generic MOACO framework is adapted to the problem at hand and explained in detail in the following section. The main disadvantages of this algorithm are that each ant in the colony either searches for low-cost options or a low percentage of late deliveries. This means that mainly the extremes of the solution space are searched. The other versions of the algorithm are introduced in Section 4.1.5 and further explained in Section 4.2 to 4.5.

4.1 Multi-objective Ant Colony Optimization algorithm

The problem definition given in Section 3.1 is easily adaptable to a graph representation. We define two sets of nodes, one set N for the items $i \in I$, one set T for the suppliers $v \in \mathcal{V}$. Furthermore, we define a set of directed arcs, A_i for each item $i \in I$. One directed arc is defined for each $v \in V_i, \forall i \in I$, starting in the item node and ending in the supplier node.

Algorithm 1 contains an overview of the g-ACO algorithm and in this section, the algorithm is explained in detail. At the initiation of the algorithm, a predefined number of ants is generated to form one colony. Each ant is randomly assigned one objective $m \in M$, which determines which objective it will try and optimize. Furthermore, each arc $a \in A_i, \forall i \in I$, is assigned two pheromone parameters, one for each objective $m \in M$. At initiation, the global non-dominated solution set S is initiated as an empty set. The global non-dominated set S contains all non-dominated solutions found by the algorithm and is the set that will be returned in the end to the decision-makers.

Algorithm 1 Pseudocode for the g-ACO algorithm.

```
Set all $\tau_{i,v}^m, \forall i \in I, v \in V_i, m \in M$ to $\tau_{MAX}$
Create the colony and select a random objective $m \in M$ for each ant.
while the number of generations is smaller than $\phi$ do
    for each ant in the colony do
        for each item $i \in I$ do
            Assign each supplier $v \in V_i$ probability $p_{i,v}^m$ according to Equation (16).
            Select a supplier from the set $V_i$ with probability $p_{i,v}$
        end
        Construct a solution.
        Check the feasibility of the solution
    end
    Find the solutions with the best value for each objective $m \in M$.
    for Arc $(i, v) \in A$ do
        Perform pheromone evaporation according to Equation (14).
        if $\tau_{i,v} < \tau_{MIN}$ then
            Set $\tau_{i,v}$ to $\tau_{MIN}$.
        end
        if $\tau_{i,v} < \tau_{MAX}$ then
            Set $\tau_{i,v}$ to $\tau_{MAX}$.
        end
        if Arc $(i, v)$ is part of one of the best solutions found then
            Update the pheromones on arc $(i, v)$ according to Equation (15)
        end
        .
    end
    Update the global non-dominated solution set.
end
```

Each generation each ant in the colony chooses one of the arcs $a \in A_i$ for each of the items $i \in I$. Which supplier is chosen for an item is determined by the transition procedure described in Section 4.1.3. There are in total ϕ generations.

Once each ant in the colony has created a solution, two ants are selected to perform pheromone depositing, explained in Section 4.1.2. These ants are the ones with either the lowest cost or the lowest fraction of late lines. Additionally, the pheromone levels for each arc in the graph evaporate according to the pheromone evaporation scheme explained in Section 4.1.2.

Once each ant in a generation has created a feasible item assignment, the global non-dominated set S is updated. The set S is initiated as empty and the first solutions are added after the first generation. As the name suggests only non-dominated solutions, which are Pareto efficient, see Section 1 for the explanation, are added to set S. Each solution generated in the current generation is checked whether it is dominated by any one of the solutions in the global non-dominated set. If it is non-dominated the solution is added to the global non-dominated set and any solutions in the non-dominated set that are dominated by the new solution are removed from the non-dominated set. Once the algorithm is finished, the non-dominated solution set S is returned.

The main disadvantage of using this algorithm is that each ant focuses solely on one objective and only the two best ants are rewarded each generation. This strategy is expected to provide solutions that lay in the extreme parts of the solution space and there will be fewer solutions in the centre of the solution space. Another disadvantage is that the algorithm is expected to converge to a solution set fast. This is the case as each generation, the ants with the same preference assign the same probability for each arc. When one arc has gotten many pheromones early in the algorithm, each generation this arc will get chosen more often and has a higher probability of being included in the best solution found in the generation.

4.1.1 Feasibility

The solution that is created by each ant is checked whether it is feasible and if needed to be adjusted to a feasible solution in regards to the MOV and MOQ of each supplier involved, using Algorithm 2. This algorithm makes a distinction between suppliers with a lead-time that is shorter or equal to 28 days and suppliers with a higher lead-time than 28 days to factor in lead-time. Afterwards three different algorithms, namely Algorithms 3, 4 and 5, can be executed where the next algorithm is executed only when the previous algorithm did not result in a feasible result for the supplier under investigation. The order in which these algorithms are used is determined by the lead-time. If the supplier accepts a penalty when the MOQ or MOV is not met and the cost of the penalty is smaller than the cost increase of a new solution, the original solution for that supplier is still used and the penalty will be accepted by default.

Algorithm 2 Pseudocode for making an infeasible solution feasible.

```
for $v \in V$ do
    if supplier $v$ did not attain their MOV or MOQ then
        Define $W$ as the set of used suppliers in the solution and $J_w$ as the items assigned to supplier $w \in W$.
        if $l_v \leq 28$ days then
            Perform Algorithm 4.
            if algorithm 4 did not provide a feasible solution for supplier $v$ then
                perform Algorithm 3.
                if algorithm 3 did not provide a feasible solution then
                    Perform Algorithm 5
                end
            end
        else
            Perform Algorithm 3.
            if algorithm 3 did not provide a feasible solution for supplier $v$ then
                Perform Algorithm 4.
                if algorithm 4 did not provide a feasible solution then
                    Perform Algorithm 5
                end
            end
        end
    end
    if supplier $v$ accepts a penalty $p$ then
        if the cost of the new assignment for supplier $v$ is higher than $p$ then
            The original assignment for supplier $v$ is used ant the penalty accepted.
        end
    end
end
```

Algorithm 3 redistributes all items assigned to the supplier which did not attain their MOV or MOQ to new suppliers. The suppliers that can be considered for an item reassignment are the suppliers that already have at least one item assigned in the current solution and they must be able to supply the item. The supplier chosen for an item is the one with the lowest per piece price. This algorithm will not lead to a new feasible allocation when an item can only be supplied by the supplier that did not attain their MOV or MOQ, as this item cannot be reallocated in that case.

Algorithm 3 Pseudocode for redistributing items over suppliers.

```
for $i \in J_v$ do
    Define $new$ as the new supplier for item $i$ and initiate as $None$.
    Set $C_{i,None}$ to $\infty$
    for $w \in I_v$ do
        if $w \in W$ and $w \neq v$ then
            if $c_{i,w} < c_{i,new}$ then
                Set $new \leftarrow w$.
            end
        end
    end
    if $new = None$ then
        Return the original solution for supplier $v$.
        Break
    else
        Add item $i$ to $J_{new}$.
    end
end
Recalculate the cost and lead-time for the new solution and for each supplier $w \in W$.
Remove supplier $v$ from $W$.
Return the updated solution.
```

Algorithm 4 Pseudo-code for increasing the number of different items assigned to supplier v.

```
Define $I_v \subset I$ as the set of items such that $v \in V_i$ and $i \notin J_v$.
Sort set $I_v$ by decreasing value of $c_{i,v} \cdot q_i$.
for Item $i \in I_v$ do
    if reassigning item $i \in J_w$ to supplier $v$ does not make supplier $w$ infeasible then
        Reassign item $i$ to supplier $v$
        Recalculate the costs and leadtime for both supplier $v$ and $w$.
        Add $i$ to set $J_v$ and remove from $J_w$. if The solution is feasible w.r.t supplier $v$ then
            break
        end
    end
end
if the final reassignment is feasible w.r.t supplier $v$ then
    Return the new solution.
else
    Return the original solution
end
```

Algorithm 4 tries to reallocate one (or multiple) item from another supplier w to the supplier v which did not attain their MOV or MOQ, without making the allocation infeasible for the first supplier w. Items that qualify for this reassignment can be supplied by the supplier under investigation and the reassignment must not create a MOV or MOQ infeasibility for the supplier from which the items are taken. This algorithm will not lead to a new feasible allocation when there are not enough items that can be reallocated to the infeasible supplier v.

Algorithm 5 is performed last in all occasions where the previous two algorithms do not lead to a feasible solution. This happens for example when an item is only supplied by one supplier so it cannot be redistributed from the supplier that did not attain their MOV or MOQ. This algorithm is performed last as it will always lead to a cost increase. It works as follows. All items already assigned to the supplier are proportionally increased such that the MOQ or MOV is met. This algorithm will always provide a new allocation that is feasible. However, it might provide a solution with very high costs in which case the solution will never be included in the global non-dominated set.

Algorithm 5 Pseudocode for increasing the number of items for supplier v.

```
Set $atV = True$ if supplier $v$ has attained their MOV and $atV = False$ otherwise.
Set $atQ = True$ if supplier $v$ has attained their MOV and $atQ = False$ otherwise.
Define $\Gamma$ as the total quantity already assigned to supplier $v$.
Define $\Omega$ as the total monetary value already assigned to supplier $v$.
for $i \in J_v$ do
    Define $\gamma$ as the total quantity of item $i$ already assigned to supplier $v$
    Define $\omega$ as the total monetary value of item $i$ already assigned to supplier $v$
    if $atQ = False$ then
        Increase the quantity of item $i$ with $\lfloor \frac{\gamma}{\Gamma} \cdot (MOQ - \Gamma) \rfloor$
    end
    Reevaluate $atV$.
    if $atV = False$ then
        Increase the quantity of item $i$ such that it has a value of more or less $\frac{\omega}{\Omega} \cdot (MOV - \Omega)$, depending
        on $u_{i,v}$.
    end
    Reevaluate $atQ$
end
Return the new updated solution for supplier $v$.
```

4.1.2 Pheromone structures and updating

Each of the arcs $(i, v) \in A_i$ for each of the items $i \in I$ is assigned two pheromone parameters, one for each objective $m \in M$, defined as $\tau^m_{i,v}$ and at initialization set to τ_{MAX}. This parameter indicates whether the arc has previously been included in good solutions and cannot become higher than τ_{MAX} or lower than τ_{MIN}. Each generation the pheromone parameters for each objective $m \in M$ for each arc (i, v) in the graph evaporate following Equation (14). This evaporation ensures that the arcs included in recent generation best solutions are favored over arcs included in less recent generation best solutions.

$$\tau^m_{i,v} = (1 - \theta) \cdot \tau^m_{i,v} \tag{14}$$

where θ is a hyper-parameter determining how much pheromone should evaporate each generation.

The ant with the best solution regarding the fraction of late lines and the ant with the best solution regarding the total cost (these may be the same ant) are allowed to increase the pheromone parameter regarding its objective m on the arcs (i, v) included in that solution with an amount of $\Delta\tau^m_{i,v}$, following

Equation (15).

$$\Delta\tau_{i,v}^{m} = \frac{1}{1 + f_m(x^m) - f_m(x_{best}^{m})}. \tag{15}$$

where $f(x_{best}^{m})$ is the value of objective m of best solution over all generations of the algorithm including the current generation regarding objective m and $f(x^m)$ is the value of objective m of the best solution regarding objective m of the current generation.

4.1.3 Transition procedure

Each ant in the colony needs to choose one supplier from the set $v \in V_i$ per item $i \in I$ during each generation of the algorithm. This selection is based on how desirable a supplier v is for an item i, which is represented through both the level of the pheromone parameter for arc (i, v) corresponding to the objective m, where $m \in M$ is randomly assigned to the ant at initiation, and the heuristic parameter for arc (i, v) corresponding to that same objective m. The heuristic parameter is explained in Section 4.1.4 The desirability of the arc determines the probability it will be chosen, following Equation (16).

$$p_{i,v}^{m} = \frac{[\tau_{i,v}^{m}]^{\alpha} \cdot [\eta_{i,v}^{m}]^{\beta}}{\sum\limits_{l \in V_i} [\tau_{i,l}^{m}]^{\alpha} \cdot [\eta_{i,l}^{m}]^{\beta}}. \tag{16}$$

where α represents the relative importance of the pheromone level of the arc and parameter β represents the relative importance of the heuristic parameter of the arc.

4.1.4 Heuristic parameters

For each arc in the graph there will be a heuristic value defined for each objective. This heuristic value represents the quality of the choice in terms of lead-time, $\eta_{i,v}^{l} \forall i \in I, v \in v_i$ and per piece price of an item, $\eta_{i,v}^{c} \quad \forall i \in I, v \in v_i$. The formulas are defined in Equation (17) and (18), respectively.

$$\eta_{i,v}^{t} = \min\{1, \max\{0 + \varepsilon, \frac{L^{max} - t_{i,v}}{L^{max} - L^{min}}\}\} \qquad \forall i \in I, v \in V_i \tag{17}$$

$$\eta_{i,v}^{c} = \min\{1, \max\{0 + \varepsilon, \frac{C_i^{max} - c_{i,v}}{C_i^{max} - C_i^{min}}\}\} \qquad \forall i \in I, v \in V_i \tag{18}$$

Here L^{max} and L^{min} are the maximum and minimum lead-time over all suppliers present in a set $V_i, \forall i \in I$ respectively. Parameters C_i^{max} and C_i^{min} represent the maximum and minimum per piece price of an item and the maximum and minimum are not taken over the entire set of relevant suppliers, but over the set of suppliers V_i per item i. This is because the per piece price can vary greatly between items. The purchase price of a car will be significantly higher than that of a connection cable or a pack of aspirins. We decided to not separate the lead-time parameters L^{max} and L^{min} as the lead-times of different suppliers and different items will not be drastically different, as they are all measured in days.

4.1.5 Other algorithm versions

One extension following the same set-up as the g-ACO algorithm is considered. This second algorithm is based on the original work of Dorigo & Gambardella (1997) and inspired by the multi-objective adaption by Yang et al. (2010). This version of the algorithm will be called t-ACO (traditional ACO) and has two advantages compared to the g-ACO. In the t-ACO algorithm, every ant in the algorithm uses different weights for each objective (summed the weights add up to 1), which means the entire solution space is searched. Another advantage compared to the g-ACO is that this algorithm uses a pseudo-random rule which determines at each step of the algorithm whether the ant prefers exploration or exploitation.

Furthermore, three complex adaptations are explained. These adaptions are based on the work of three different authors. These three methods are all fairly new proposals that have provided very promising results as mentioned in the taxonomy by Falcón-Cardona et al. (2020). What these methods have in common is that they use the same elements as in more common used MOACO algorithms, yet in more complex ways to overcome disadvantages such as having to update a whole set of solutions each generation to find the non-dominated ones or biased pheromone updating rules.

The first of these three algorithms is the adaption of the work by Mora et al. (2013), which we will call the CHAC. This algorithm employs multiple sub-colonies as opposed to the t-ACO and g-ACO which both use one colony. The sub-colonies each have unique preference weights determining which objective they prefer and how much. Each sub-colony employs a pheromone structure and non-dominated solution set. This has the advantage that after each generation for each sub-colony a smaller set of non-dominated solutions has to be updated, resulting in a faster algorithm. Another advantage is that each sub-colony can be allowed to search for solutions in parallel, as the ants in sub-colonies do not communicate. A disadvantage is that, as there are two pheromone parameters per arc per sub-colony, more memory is needed to store these parameters.

The second of these three is an adaption of the work by Angus (2007), which will be called c-PACO (crowding population-based ACO). This algorithm has two main differences from all the algorithms seen so far. First, the size of the non-dominated solution set is determined as a hyper-parameter and only a subset of those solutions is checked against a new solution. The advantage of this is that in the case there are many non-dominated solutions the algorithm does not slow down because many solutions need to be compared. A disadvantage, however, is that as the size of the solution set is predetermined, we might not return a good set of non-dominated solutions, which is the goal of this research. The second difference is the way pheromones are updated. The entire set of solutions found in one generation is ranked and used to update the pheromones, leading to less bias compared to the t-ACO and g-ACO algorithms.

The third and last of these three more complex methods are based on the work of Afshar et al. (2009) and will be called na-ACO (non-dominated archiving ACO). The sequence of events happening in this algorithm is very different from the other algorithms introduced so far. The algorithm uses two independent colonies, each only searching for solutions that optimize one objective. The solutions found by one colony are evaluated by the other colony to perform pheromone updates. Furthermore, the global set of non-dominated solutions called the archive in this case, is not updated every generation, resulting in a faster algorithm.

4.2 Traditional Ant colony optimization algorithm

In this section, we will discuss an algorithm based on the MOACO employed in the work of Yang et al. (2010). The work is derivative of the original single-objective ACO proposed by Dorigo & Gambardella (1997), which is the reason we will be calling this method the traditional MOACO (t-ACO) in this research. Yang et al. (2010) apply their ACO algorithm to a grid over optical burst switching networks problem, or they need to assign computational resources to jobs to minimize completion time and perform load balancing. The problem investigated in their research is analogous to the problem at hand, as the problem structure is similar, a list of resources needs to be assigned to a list of jobs. An overview of the algorithm employed in this research can be found in Algorithm 6.

Algorithm 6 Pseudocode for the t-ACO algorithm.

```
Set all $\tau_{i,v}^{m}, \forall i \in I, v \in V_i, m \in M$ to $\tau_{MAX}$
while The number of generations is smaller than $\phi$ do
    for each ant in the colony do
        for each item $i \in I$ do
            Assign each supplier $v \in V_i$ probability $p_{i,v}^{m}$ according to Equation (22).
            Select a supplier from the set $V_i$ according to the pseudo-random proportional rule.

        end
        Construct the solution.
        Check the feasibility of the solution.
    end
    for Arc $(i, v) \in A$ do
        Perform pheromone evaporation according to Equation (19).
        if $\tau_{i,v} < \tau_{MIN}$ then
            Set $\tau_{i,v}$ to $\tau_{MIN}$.
        end
        if $\tau_{i,v} < \tau_{MAX}$ then
            Set $\tau_{i,v}$ to $\tau_{MAX}$.
        end
        if Arc $(i, v)$ is part of one of the best solutions found then
            Update the pheromones on arc $(i, v)$ according to Equation (20)
        end
        .
    end
    Update the global Pareto solution set.
end
```

In this version of the algorithm, the same sequence of procedures occurs as in the g-ACO algorithm described in the previous section. However, the pheromone updating and transition procedures are slightly different and will be explained in the following sections.

4.2.1 Pheromone structures and updating

The t-ACO algorithm utilizes only one pheromone parameter per arc $(i, v) \in A_i$ for each item $i \in I$ defined as $\tau_{i,v}$. The pheromone parameter is initialized to τ_{MAX} as in the g-ACO algorithm, and may not exceed τ_{MAX}

or fall beneath τ_{MIN}. In this case the parameter indicates whether the arc (i, v) has been part of solutions belonging to the non-dominated set. Each generation the pheromone parameters evaporate according to Equation (19).

$$\tau_{i,v} = (1 - \theta) \cdot \tau_{i,v} \tag{19}$$

In the t-ACO all the ants with solutions belonging to the global non-dominated set are allowed to deposit pheromones on the arcs included in their solution. The amount of pheromone deposits follows the Equation as described in Equation (20).

$$\Delta\tau^m_{i,v} = \frac{1}{1 + f_m(x^m) - f_m(x^m_{best})}. \tag{20}$$

where $f(x^m_{best})$ is the value of objective m of best solution over all generations of the algorithm including the current generation regarding objective m and $f(x^m)$ is the value of objective m of the best solution regarding objective m of the current generation.

A disadvantage of this method is that the pheromones are updated by the non-dominated set, meaning that each generation, there is more convergence towards the solutions that are already in the non-dominated set. This will limit exploration in the long run and increase the chance of getting stuck in a local optimum.

4.2.2 Pseudo-random proportional transition procedure

Yang et al. (2010) employ the *pseudo-random proportional rule* to decide which arc to walk over next. This rule is used to diversify the search through differentiation between exploitation and exploration each step of the algorithm. Define a threshold value q_0 on the interval (0,1). Each time an ant needs to make a transition decision, a random value q in the range of [0,1] is drawn. If $q < q_0$ the ant will prefer exploitation and otherwise it will prefer exploration. The probabilities of each supplier $v \in V_i, \forall i \in I$ are defined in Equations (21) and (22).

If $q < q_0$:

$$p_{i,v} = \begin{cases} 1, & \text{if } v = \arg\max_{v \in V_i}\{[\tau_{i,v}]^\alpha \cdot [[\eta^c_{i,v}]^\lambda \cdot [\eta^l_{i,v}]^{(1-\lambda)}]^\beta\} \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

else:

$$p_{i,v} = \frac{[\tau_{i,v}]^\alpha \cdot [[\eta^c_{i,v}]^\lambda \cdot [\eta^l_{i,v}]^{(1-\lambda)}]^\beta}{\sum\limits_{l \in V_i} [\tau_{i,l}]^\alpha \cdot [[\eta^c_{i,l}]^\lambda \cdot [\eta^l_{i,l}]^{(1-\lambda)}]^\beta} \tag{22}$$

where λ is a parameter that determines which objective the ants prefer. In their research, Yang et al. (2010) assign a value of 0.5 to λ for each ant, meaning that the two objectives are equally important. This will result in the ants mainly exploring towards the centre of the solution space and not towards the extreme points. When λ is larger than 0.5 the ant will prefer to minimize the cost objective, while when λ is smaller than 0.5 the ant will prefer to minimize the total fraction of late lines.

4.3 Compañía de Hormigas ACorazadas

In this section, a MOACO algorithm based on the work by Mora et al. (2013) is presented. With their work, they aim to improve the quality of the non-dominated solution set by subdividing a colony into N sub-colonies, in contrast to the two algorithms already explained where only one colony was employed. The ants from each colony search in a specific part of the solution space, to provide a diverse and high-quality non-dominated solution set. They find that this *island* method works best with the Compañía de Hormigas ACorazadas (CHAC) algorithm (Mora et al., 2009). The outline of this algorithm can be found in Algorithm 7 and a flow representation can be found in Appendix B, Figure 9.

Algorithm 7 Pseudocode for the CHAC algorithm.

```
for $c \in [1, N_c]$ do
    Determine $\lambda_c$ using Equation (28).
    Set all $\tau^m_{i,v}(c), \forall i \in I, v \in V_i, m \in M$ to $L_m$.
end
while the number of generations is smaller than $\phi$ do
    for $c \in [1, N_c]$ do
        for each ant in the colony with number $c$ do
            for each item $i \in I$ do
                Assign each supplier $v \in V_i$ probability $p_{i,v}$ according to Equation (27).
                Select a supplier $v \in V_i$ following the PS.
                Perform local pheromone update on arc $(i, v)$ following Equation (23) Construct the solution.
                Check the feasibility of the solution.
            end
            Update the Pareto solution set of the $c$th colony.
        end
        if a random number on interval $[0, 1] < 0.1$ then
            Perform migration following Algorithm 8
        end
        for $c \in [1, N_c]$ do
            Perform global pheromone updating for the $c$th colony following Equation (24). Let the ants in the pareto-optimal set of colony $c$ lay down pheromones on their solution path following Equation (25).
        end
    end
    Add the Pareto set of each sub-colony to one set and determine the non-dominated solutions.
end
```

4.3.1 Pheromone structures and updating

This algorithm employs two pheromone parameters per sub-colony $s \in N$ for each arc $(i, v) \in A_i, \forall i \in I$, one for each objective under consideration. In total there will be $2 \cdot |N|$ pheromone parameters per arc. The pheromone parameters connected to two different colonies are independent. The amount of pheromone trail for each objective for each colony $c \in N_s$ is initiated to τ_{max} as in the previous two sections. Again, the pheromone level can not be higher than τ_{max} or lower than τ_{min}.

After the ant from sub-colony $c \in N_s$ has chosen a supplier $v \in V_i$ for an item $i \in I$, the amount of pheromone trail $\tau^m_{i,v}(c)$ on arc (i, v) is updated following Equation (23). This procedure prevents the sub-colony $c \in N_s$ from getting stuck in a local optimum. As each ant in the sub-colony has the same preference weights regarding the two objectives, when this procedure would not be used, the probability of choosing a certain supplier for a certain item would be the same for each ant. This gives a higher probability that similar solutions are found many times in an generation, leading to fast convergence of the algorithm. Using evaporation after each choice of an ant in the sub-colony leads to more diverse solutions and slower convergence.

$$\tau^m_{i,v}(c) = (1-\theta) \cdot \tau^m_{i,v}(c) + \theta \cdot \tau_0 \tag{23}$$

The non-dominated solution set is updated for each sub-colony after each generation. Pheromone updates in this scheme happen at the end of each generation for each separate sub-colony, after migration. The evaporation formula is the same as in previous sections, Equation (24), and the pheromone is laid on the edges belonging to solutions in the Pareto set for both objectives, following Equation (25).

$$\tau^m_{i,v}(c) = (1-\theta) \cdot \tau^m_{i,v}(c) \qquad \forall m \in [0,M], c \in N_s i \in I, v \in V_i \tag{24}$$

$$\Delta\tau^m_{i,v}(c) = \frac{1}{f_m(x)} \qquad \forall m \in [0,M], c \in N_s, i \in I, v \in V_i \tag{25}$$

where $f(x^m_{best})$ is the value of objective m of best solution over all generations of the algorithm including the current generation regarding objective m and $f(x^m)$ is the value of objective m of the best solution regarding objective m of the current generation.

4.3.2 Transition procedure

The transition decision rule used for this algorithm is the same as in Section 4.2. The equations used within this pseudo-random proportional transition rule are given in Equation (26) and (27). Equation (26) is employed when $q < q_0$ otherwise a random supplier is chosen based on the proportional probabilities given in Equation (27).

If $q < q_0$:

$$p_{i,v} = \begin{cases} 1, \text{ if } v = \arg\max_{v \in V_i} \{[\tau^1_{i,v}]^{\alpha\lambda_s} \cdot [\tau^2_{i,v}]^{\alpha(1-\lambda_s)} \cdot [\eta^1_{i,v}]^{\beta\lambda_s} \cdot [\eta^2_{i,v}]^{\beta(1-\lambda_s)}\} \\ 0, \text{ otherwise} \end{cases} \tag{26}$$

else:

$$p_{i,v} = \frac{[\tau^1_{i,v}]^{\alpha\lambda_s} \cdot [\tau^2_{i,v}]^{\alpha(1-\lambda_s)} \cdot [\eta^1_{i,v}]^{\beta\lambda_s} \cdot [\eta^2_{i,v}]^{\beta(1-\lambda_s)}}{\sum\limits_{l \in V_i} [\tau^1_{i,l}]^{\alpha\lambda_s} \cdot [\tau^2_{i,l}]^{\alpha(1-\lambda_s)} \cdot [\eta^1_{i,l}]^{\beta\lambda_s} \cdot [\eta^2_{i,l}]^{\beta(1-\lambda_s)}} \tag{27}$$

Where $\lambda_s \in [0,1]$ is determined by the sub-colony $s \in N$ the ant comes from. Then, λ_s is determined following Equation (28). When λ_s is larger than 0.5, the ant prefers paths with a low cost and when λ_s is smaller than 0.5 it prefers paths with low lead-time. The novelty from this method is that multiple ants use the same value of λ_s such that they search the same space, instead each ant using a different value for λ (Iredi et al., 2001).

$$\lambda_s = \frac{a-1}{|N|-1}, \qquad \forall s \in N \tag{28}$$

Where $a \in [1, |N|]$ is the number of sub-colony s.

4.3.3 Migration between colonies

In this method, ants are allowed to migrate to another sub-colony in randomly chosen generations. The outline of the migration procedure can be found in Algorithm 8. The ant that will migrate to another sub-colony is one from the Pareto set with the best objective according to the sub-colony it will migrate to. The sub-colonies with a value of λ_s higher than 0.5 will always migrate to a sub-colony with a higher λ_s, resulting in the ant with the lowest cost migrating. The sub-colonies with a value of λ_s lower than 0.5 will always migrate to a sub-colony with a smaller λ_s, resulting in the ant with the lowest percentage of late items migrating. The ants from sub-colonies with a value of λ_s equal to 0 or 1 will never migrate, these sub-colonies will only receive ants. If there is a sub-colony with λ_s equal to 0.5, this sub-colony will let two ants migrate. After the migration, the Pareto sets are updated for each separate sub-colony.

Algorithm 8 Algorithm for migration in the Pareto-based Island model.

```
for each colony $c \in [1, |N_c|]$ do
    if $\lambda_c < 0.5$ and $\lambda_c \neq 0$ then
        Migrate the solution with lowest cost toward the Pareto solution set of colony $c - 1$.
        Update the Pareto solution set of colony $c - 1$.
    end
    if $\lambda_c > 0.5$ and $\lambda_c \neq 1$ then
        Migrate the solution with lowest percentage late items toward the Pareto solution set of colony
        $c + 1$.
        Update the Pareto solution set of colony $c + 1$.
    end
    if $\lambda_c = 0.5$ then
        Migrate the solution with lowest cost toward the Pareto solution set of colony $c - 1$.
        Update the Pareto solution set of colony $c - 1$.
        Migrate the solution with the lowest percentage late items toward the Pareto solution set of colony
        $c + 1$.
        Update the Pareto solution set of colony $c + 1$.
    end
end
```

4.4 Crowding population based ACO

In this section, we describe a MOACO algorithm called crowding population-based Ant Colony Optimization (c-PACO) developed by Angus (2007). This multi-objective optimization algorithm is based on the original single-objective population-based ACO algorithm (Guntsch & Middendorf, 2002). The c-PACO is similar to the g-ACO in the sense that the sequence of events occurring is similar. However, the methods used for pheromone updating and determining the non-dominated solution set are completely different. The advantage of this ACO algorithm over others is the speed improvement obtained by solution storage operations. PACO uses the population of ants to make immediate changes to the pheromone matrices. As it is important in this research to obtain a good result within a reasonable time, we will consider this algorithm alongside the others proposed. The outline of the algorithm can be found in Algorithm 9 and a flow representation can be found in Appendix B, Figure 10.

Algorithm 9 Pseudo-code for the crowding population based MOACO algorithm.

```
Define a solution population $S$ of predetermined size and fill with randomly generated solutions.
Set the pheromones on each trail to $\tau_0$.
for each ant $h$ do
    Generate a random $\lambda_h$.
end
while The number of generations is smaller than $\phi$ do
    for each ant in the colony do
        for item $i \in I$ do
            Determine the probability $p_{i,v}$ for each supplier $v \in V_i$ following Equation (29).
            Select a random supplier based on their probabilities $p_{i,v}$
        end
        Construct the solution.
        Check the feasibility of the solution.
    end
    Select a random subset of solutions $S' \subset S$ and compare each new solution found in this generation
    with $S'$ using Algorithm 11.
    Rank each solution in $S$ using Algorithm 10.
    Reinitialize the pheromone trail on each arc to $\tau_0$.
    for each solution $s \in S$ do
        Increase the pheromone trail on each arc $(i, v)$ in solution $s$ by $\frac{1}{\text{rank}}$.
    end
end
Determine which solutions in $S$ are non-dominated
```

4.4.1 Pheromone structures and updating

The c-PACO uses one colony and one pheromone parameter per arc $(i, v) \in A_i, \forall i \in I$ per generation. At the end of each generation, a new pheromone matrix is constructed in the following way. Firstly, all the solutions in S are given an integer rank using Algorithm 10, based on the famous work of Deb et al. (2000). Secondly, all elements in the pheromone matrix are initialized to τ_0. Lastly, all solutions s in population S increase the pheromones corresponding to the arcs in that solution with $\frac{1}{s_{rank}}$.

Algorithm 10 is often referred to as the fast non-dominated sorting ranking algorithm. The goal of the algorithm is to rank all the solutions found in an generation as fast as possible. The algorithm does this by creating multiple fronts of solutions that are amongst them self non-dominated. The true non-dominated set will get the rank of 1 and the set that is dominated by the true non-dominated set, but not amongst themselves will get rank 2, and so on.

Algorithm 10 Fast non-dominated sorting ranking.

```
for  $s \in S$ do
    $P_s =$
    $n_s = 0$
    for $q \in S \backslash \{s\}$ do
        if $s$ dominates $q$ then
            $P_s = P_s \cup \{q\}$
        end
        if $q$ dominates $s$ then
            $n_s = n_s + 1$
        end
        if $n_s == 0$ then
            $s_{rank} = 1$
            $\mathcal{F}_1 = \mathcal{F}_1 \cup \{s\}$
        end
    end
end
$i = 1$
while $\mathcal{F}_i$ do
    $Q =$
    for  $s \in \mathcal{F}_i$ do
        for $q \in P_s$ do
            $n_q = n_q - 1$ if $n_q = 0$ then
                $q_{rank} = i + 1$
                $Q = Q \cup \{q\}$
            end
        end
    end
    $i = i + 1$
    $\mathcal{F}_i = Q$
end
```

4.4.2 Transition procedure

In this MOACO the transition decision is based on the same proportional rule as in Section 4.1, with another equation for determining the probabilities for each arc (i, v) , $\forall i \in I, v \in V_i$ described in Equation (29). This means that each arc (i, v) $\forall v \in V_i$ is assigned a probability and an arc is randomly chosen amongst those with that probability.

$$p_{i,v} = \frac{[\tau_{i,v}]^{\alpha} \displaystyle\prod_{d=1}^{h} [\eta_{i,v}^{d}]^{\lambda_d \beta}}{\displaystyle\sum_{l \in V_i} [\tau_{i,l}]^{\alpha} \prod_{d=1}^{h} [\eta_{i,l}^{d}]^{\lambda_d \beta}} \tag{29}$$

Each ant in the colony is assigned a random value for parameter λ, which allows the ants to use one pheromone matrix while still exploring the entire feasible region. Parameter λ determines the weight of both objectives for a certain ant. When λ is greater than 0.5 the ant prefers minimizing the costs, while if λ is smaller than 0.5 the ant prefers minimizing the fraction of late lines. The generation of λ for each ant in the colony is determined by generating a random value on the interval $[0, 1]$.

4.4.3 Updating of the non-dominated solution set

The c-PACO uses one population of solutions S of predetermined size. After each generation, a new set of solutions Y is generated and compared to a random set $S' \subset S$. For each new solution, $s_n \in Y$ from the current generation, the closest match in S' is determined by selecting the solution with the highest number of arcs in common. The new solution s_n only replaces their closest match if s_n dominates the other. This method, called the crowding replacement scheme can be found in Algorithm 11.

Algorithm 11 Crowding replacement scheme.

```
for $s_n \in Y$ do
    Randomly select $S' \subset S$.
    Find in $S'$ the solution $s$ with largest number of edges in common with $s_n$.
    if $s_n >> s$ then
        Remove $s$ from $S$.
        Add $s_n$ to $S$.
    end
end
```

4.5 Non-dominated archiving multi-colony ant algorithm

In this section, an algorithm based on the non-dominated archiving multi-colony ant (na-ACO) algorithm proposed by Afshar et al. (2009) is introduced. The algorithm was designed to combat the highly convex multi-purpose reservoir operation. The algorithm uses elements from both the g-ACO and t-ACO. Namely, it uses two colonies and two pheromone matrices. Each ant solely prefers one objective as in the g-ACO, yet in the g-ACO this is decided randomly and in the na-ACO this is determined by the colony the ant is from. The na-ACO employs the same pseudo-random proportional rule to determine the transition decision as in the t-ACO. To overcome the disadvantages of ants solely preferring one objective, which leads to solutions found mainly in the extreme regions of the solution space, some special mechanisms are included in the na-ACO. The outline of the algorithm can be found in Algorithm 12 and a flow representation can be found in Appendix B, Figure 11.

4.5.1 Sequence of events

The sequence of events happening in the na-ACO is quite different from those in the algorithms explained thus far. This algorithm employs one colony for each objective function with one pheromone matrix per colony. Unlike the other algorithms explained so far, the non-dominated solution set is not updated after each construction. In na-ACO each ant from each colony constructs multiple solutions per generation. In the first cycle of each generation, each ant from one arbitrarily chosen colony constructs a solution based on their pheromone matrix, these solutions are added to the solution set S_{iter}, which stores all solutions found in one generation. From all these solutions the ant with the best solution based on their objective is allowed to update the pheromones. Then a new set of solutions is constructed by this same colony and added to S_{iter}, yet this time the best solution is selected by the second colony and the second colonies pheromone matrix is updated after which they construct a set of solutions which is evaluated by the first colony.

Algorithm 12 Pseudocode for the non-dominated archiving MOACO algorithm.

```
for each objective $m \in M$ do
    Initiate one pheromone structure, with one trail for each arc $(i, v)$
end
Define $S_{archive} = \emptyset$ as the set of archived non-dominated solutions.
while Thetnumber of generations is smaller than $\phi$ do
    Define $S = \emptyset$ as a set of solutions.
    Define $S_{iter}$ as all solutions found in this generation. for each ant in colony 1 do
        for item $i \in I$ do
            Assign each supplier $v \in V_i$ probability $p^m_{i,v}$ according to Equation (33).
            Select a supplier from the set $V_i$ according to the pseudo-random proportional rule.
        end
        Add the solution found to $S$ and $S_{iter}$.
    end
    while the maximum number of cycles has not been reached do
        for colony $k \in M$ do
            According to the objective of colony $k$ find the best solution $s$ in $S$. Update the pheromone
            trails on each arc $(i, v)$ included in $s$ following Equation (30).
            Set $S =$ for each ant in colony $k$ do
                for item $i \in I$ do
                    Assign each supplier $v \in V_i$ probability $p^m_{i,v}$ according to Equation (33).
                    Select a supplier from the set $V_i$ according to the pseudo-random proportional rule.
                end
                Add the solution found to $S$ and $S_{iter}$.
            end
        end
    end
    Add all non-dominated solutions from $S_{iter}$ to $S_{archive}$.
    Reset all pheromone trails for each objective to $\tau_0$. for solution $s \in S_{archive}$ do
        for item $i \in I$ do
            Add $\Delta\tau_{i,v,k}$ following Formula (31) to the selected supplier $v \in V_i$.
        end
    end
end
```

This continues a predetermined number of times, after which the entire set of solutions is evaluated and the set of non-dominated solutions is evaluated against the global non-dominated set, called the Pareto Archive, and updated after the next generation again. The advantage of the Pareto Archive is that it is updated less frequent, speeding up the run-time of the algorithm. After this, the pheromone matrices of both colonies are reset to τ_0 and updated based on the archived non-dominated set by adding $\Delta\tau_{i,v,k}$ following Equation 25 and a new generation is started.

4.5.2 Pheromone structures and updating

The starting pheromone levels are all initiated to τ_{max}, where the pheromone levels can not be higher than τ_{max} or lower than τ_{min}. A colony does not evaluate their solutions based on its objective, but the other colony does. This means that the solutions of colony 1 are used by colony 2 to update the pheromone parameters of colony 2 and vice versa. Because of this method, the algorithm is less likely to converge very fast to solutions in the extreme parts of the solution space, which does happen with the g-ACO algorithm. During an generation, the pheromone update is performed using Equation (30).

$$\tau_{i,v,k} = (1-\theta)\cdot\tau_{i,v,k} + \theta\cdot\Delta\tau_{i,v,k} \tag{30}$$

$$\Delta\tau_{i,v,k} = \begin{cases} \dfrac{L_k}{f^k(s)}, & \text{if item } i \text{ is assigned to supplier } v \text{ in solution } s, \\ 0, & \text{otherwise.} \end{cases} \tag{31}$$

Here $\Delta\tau_{i,v,k}$ is calculated using Equation (31) and $f^k(s)$ is the objective value of the best solution determined by colony k. Where L_1 is the cost objective found using a greedy heuristic aimed toward minimizing costs and L_2 is the time objective found using a greedy heuristic aimed towards minimizing lead-time. The greedy heuristic aimed at minimizing costs picks for each item the supplier with lowest per piece price. The greedy heuristic is aimed at minimizing lead-time picks for each item the vendor with the shortest lead-time.

4.5.3 Transition procedure

The transition rule used by the ants is again the pseudo-random proportional rule as explained in Section 4.2 except for that the formula used by the ants from a colony only include information about their own assigned objective and are the same as those proposed in the original work of Dorigo & Gambardella (1997), see Equations (32) and (33) in which $k \in M$ indicates the colony. This means that a colony only uses the heuristic values and pheromone parameters connected to the objective that the colony needs to optimize.

If $q < q_0$:

$$p_{i,v,k} = \begin{cases} 1, & \text{if } v = \arg\max_{v\in V_i}\{[\tau_{i,v,k}]^\alpha[\eta_{i,v,k}]^\beta\} \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

else:

$$p_{i,v,k} = \frac{[\tau_{i,v,k}(t)]^\alpha[\eta_{i,v,k}]^\beta}{\sum\limits_{l\in V_i}[\tau_{i,l,k}]^\alpha[\eta_{i,l,k}]^\beta} \tag{33}$$

5 Results

In this section, the proposed algorithms are tested for the problem at hand. Firstly, the data used for determining hyper-parameters and stochastic parameters and final testing is described. Secondly, the performance measures used to evaluate the performance of the algorithm are explained. Thirdly, which hyper-parameters need to be set to a value and how this is done is explained. Lastly, the proposed methods are tested and evaluated.

Python version 3.8 was used to implement the proposed algorithms. The computer on which the programs were run has 8.00 GB RAM-memory and a 2.2 GHz processor.

5.1 Data

The data-set used is provided by Médicins sans Frontières (MSF, 2021) exclusively for this research and consists of real-life historical orders and purchases. To be precise, the data-set consists of the following subsets: A data set consisting of 2756 historical orders from July 2019 to March 2021, a data set containing information on all available items, vendors and pricing agreements and a data set of 133567 historical purchases from October 2020 to March 2021. The difference between an order and a purchase, as defined in this research, is the following. An order consists of a list of items and the quantity of each item needed, coming directly from the MSF missions. A purchase on the other hand is the actual order placed at a vendor. A flow representation of how these different processes come together can be found in Appendix A, Figure 7. The reason the number of historical purchase orders is so much higher than the number of historical orders is that an order is usually split up into subsets of ordered items and then a subset is purchased from a specific supplier. For each of the historical purchase orders, it is known which suppliers were selected, the total AFAM costs, at which date a purchase was made and at which date the purchased items arrived at the warehouse. AFAM costs consist of service, freight and penalty costs.

In total, there are 5143 suppliers with whom MSF currently has pricing agreements. A price agreement consists of the item type, the unit measure, the time horizon for which the agreement is valid and the price per unit and the currency. There is not a price agreement available for each item, which may be due to the expiration of a price agreement. Besides the price agreements, we also know which documents a supplier can provide for each item they can deliver. Certain documents may be required for the purchased items to be allowed to be imported at destination countries. Documents may be preferred, but not required.

5.1.1 Separation train and test data

In total 312 historical orders are suitable for research. A suitable order consists of more than one item, for every item there is at least one supplier available and the date the order was placed is known. Besides, for each suitable order it is also known for each item at which supplier they where purchased and at which date the items arrived. The big difference between the number of available and suitable real-life orders stems from inconsistency in data recording, orders with only one item and outdated pricing agreements. From the 312 orders, 10 percent will be used as a test data-set and the other 90 percent as training data-set. The training set for parameter tuning of the hyper-parameters is used in the various MOACO algorithms. How this parameter tuning works is explained in Section 5.3.

To create a training-set and test-set that both contain similar orders with similar characteristics, stratification is used. This means that the entire data-set was split into separate groups, in which the orders in one group all had similar characteristics. From each of the groups 10 % was randomly selected to be included in the test-set, the remaining orders were all included in the training-set.

To perform stratification, the complete data-set of suitable orders was split on the average number of items and the average number of available suppliers. For every suitable order, the average number of available suppliers per item was calculated, which functions as a proxy for the complexity of the instance. To split the data into four groups the average number of available suppliers over the entire data-set was calculated which was 2.8. The data set was consequently split into a group with orders with less and a group with more than 2.8 available suppliers per item on average. These two groups were then split into two new groups based on the number of items in the order again based on the average number of items per order over the entire data set, which was 26.4 items per order. Each group differs in the combination of whether the average number of available vendors per order and the number of items in the order is below or above the average of the entire set of suitable orders.

The descriptive statistics of the two data-sets can be found in Table 1. It can be seen in the table that 29 orders are used for testing and 283 for training. There is a difference of 3 in the average number of items per order (29 vs 26), yet the average number of suppliers per item is very similar (2.81 vs 2.80). The test and train data-sets that resulted from the stratification method have very similar characteristics.

	Test data-set	Training data-set
Number of orders	29	283
Average number of items per order	29	26
Maximum number of items per order	226	302
Minimum number of items per order	2	2
Average number of suppliers per item	2.81	2.80
Maximum number of suppliers per item	7.44	7.14
Minimum number of suppliers per item	1.31	1.03

Table 1: Descriptive statistics of the test and training set of orders.

5.1.2 Uncertain parameter estimation

In this section, the estimation of the uncertain parameters is explained. As is explained in Section 3.1 the AFAM costs and lead-times parameters are subject to uncertainty. The historical information about AFAM costs and lead-times, obtained from the data-set consisting of historical purchases, are used to estimate the lead-time and freight costs in the model. The methods and assumptions used to obtain the estimations are explained in this section.

To estimate the expected lead-time in days, we record the average lead-time for two different metrics. The first is the average lead-time of an item when ordered from a specific supplier. The second is the average lead-time per purchase from a specific supplier, recorded from the moment the purchase was made until all items in the purchase arrived in the warehouse. This information can be obtained from the data-set consisting of historical purchases. In some cases, the date at which the purchase arrived at the warehouse was not recorded, in which case the observation was not used.

The average lead-time per item, when ordered from a specific supplier, is assumed to give the best estimation of lead-time in the future. However, in some cases, an item might not have been ordered from a specific vendor many times or even at all. In this case, the average lead-time per purchase from a specific supplier is used as the estimation of lead-time. As the average number of observations per item per supplier is 6 (see Table 2, we will use that number as a threshold. When a supplier is fairly new and less than 6 purchases were made with that supplier, taking the average lead-time of the supplier as a proxy may again be unreliable. In this case, the agreed-upon lead-time will be used as the estimation of the lead-time. This agreed-upon lead-time is recorded in the pricing agreements, which are contracts between the suppliers and Médicins sans Frontières and are included in the data-set consisting of information about items, suppliers and pricing agreements, yet in practice, this lead-time often is exceeded.

The AFAM costs for each item for each supplier are estimated identically, with the same number of observation thresholds as in the lead-time estimation. Estimating historical AFAM costs averages requires a delicate and detailed process. The assumption is made that freight costs are proportional to the number of units of an item ordered. Firstly, the average AFAM costs per item per supplier are obtained in the following way. For each historical purchase order, AFAM costs are charged. The AFAM costs are always positive. Given a historical purchase order consisting of items ordered and quantities ordered. The AFAM costs belonging to one of these items is calculated relative to the number of units of that item that were purchased. The total AFAM costs are multiplied by the number of units of the item and then divided by the total number of units purchased. Secondly, the average AFAM costs per supplier are calculated. When there are less than 6 purchases placed with a supplier, we assume total AFAM costs are 76 euro's as this is the average AFAM cost per supplier. This assumption might be an over or underestimation, but as more purchases are made the data-set containing historical data will grow and the estimations will become more accurate.

Table 2 contains descriptive statistics regarding averages of historical lead-times and AFAM costs. It can be seen from the table that there is a significant difference in the number of observations. The number of purchases available to be used to calculate the average lead-time for a specific item from a specific supplier can range from 1 to 66. The table also contains the maximum, average and minimum lead-time and AFAM costs, either calculated per item per supplier or per supplier. The minimum lead-time of an item in days is observed to be 0, while the maximum was 554, which is a year and a half. This information shows how dispersed the lead-times are. The difference in the minimum (and maximum) lead-time as calculated per item from a specific supplier or per supplier stems from the fact that the lead-time per supplier is calculated from the moment the purchase is placed until every item in the purchase has arrived in the warehouse.

		Average per observation			Number of observations		
		maximum	average	minimum	maximum	average	minimum
Per item	AFAM costs *	800	4.64	0	66	3	1
	lead-times *	554.0	65.45	1	135	6	1
Per supplier	AFAM costs *	2693.33	75.78	0	1083	41	1
	lead-times *	320.5	64.54	4	7629	158	1

* The AFAM costs are in euro's and lead-time is measured in days.

Table 2: Descriptive statistics of data-set historical lead-times and freight costs.

5.2 Performance measures

To evaluate the quality of the solution sets obtained from the methods proposed in Section 4, a performance measure is introduced in this section. The most foolproof method would be to compare an obtained Pareto set to the exact non-dominated set Jaszkiewicz (2004). However, obtaining this exact non-dominated set, also called the Pareto Frontier, is computationally expensive and as the problem at hand is NP-complete might even be impossible to obtain.

The goal of any method for finding a non-dominated solution set is to find a set that has three characteristics. The first being, minimal distance to the exact non-dominated set. The second, a maximum spread over the objective space, which means the most extreme points of the non-dominated set found are laying as far as possible from one another. The third is that the solutions are as evenly spread over the solution space as possible Knowles et al. (2006). Furthermore, the closer the non-dominated set is to the exact non-dominated front, the better the approximate set is. To assess the methods proposed in this research on those qualities, several different performance measures will be employed.

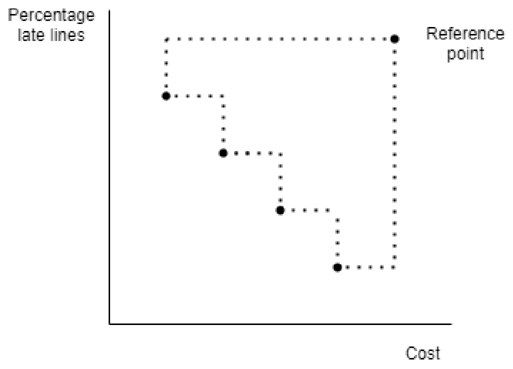
To evaluate the quality of an approximate non-dominated set obtained by an algorithm the hyper-volume indicator is utilized. The hyper-volume indicator was first introduced by Zitzler & Thiele (1999) and is used to assess the overall quality of a non-dominated solution set with one performance measure. This hypervolume indicator indicates how much of the solution space is dominated by the non-dominated solution set, which we denote by A, relative to some reference point r. The bigger the hyper-volume metrics is the better the non-dominated solution set is. This hyper-volume indicator is calculated following:

$$HV(A, r) = \lambda^* \big(\cup_{i \in A} \, [i : r] \big) \tag{34}$$

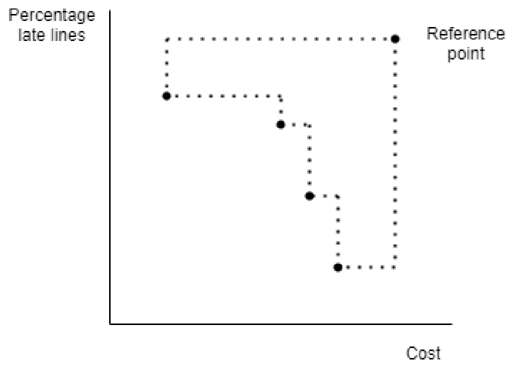
Here λ^* is the two-dimensional Lebesgue measure (Bartle, 2014). The Lebesgue measure gives the volume of a polygon based on the Euclidean space between each of the closest points in the non-dominated solution set and between the reference point and the extreme solutions. In the two-dimensional case, it measures the distance between two points along the x-axis and y-axis.

Figure 2 contains a graphical depiction of the hyper-volume metric. Figure 2a and 2b both contain a graph with four non-dominated solutions in the non-dominated set and one reference point. The reference point is in both cases chosen to have a larger cost than the most expensive solution and a higher percentage late lines than the solution with the highest fraction of late lines in the non-dominated solution set.

The non-dominated solution set depicted in Figure 2a has solutions that are equally spread and the front forms a straight line. On the contrary, the non-dominated solution set depicted in Figure 2b has solutions that are not equally spread and form a non-linear line. It can be easily seen from the graph that the hyper-volume of the non-dominated solution set depicted in Figure 2a is larger than the one in Figure 2b.



(a) Hyper-volume depiction of a solution set with solutions that are equally spaced and form a straight line.



(b) Hyper-volume depiction of a solution set with solutions that are irregularly spaced and form a non-linear line.

Figure 2: Graphical depiction of a two-dimensional hyper-volume (area). The size of the area between the dotted lines is the size of the hyper-volume.

Hyper-volume is the only known performance measure that is strictly monotonic (Audet et al., 2020), it evaluates all the three desirable traits of a non-dominated solution set and by standardizing the hyper-volume indicator it is possible to compare different instances to one another. The three desirable treats being, proximity to the exact non-dominated solution set, maximum spread over the objective space and equal spread over the non-dominated solution set. As will be explained below, the hyper-volume measure will prefer non-dominated solution sets adhering to these traits over non-dominated solution sets that do not have these traits.

How the reference point r is determined is important when multiple non-dominated solution sets are compared to another. Often the reference point is chosen to be slightly worse than the nadir-point (Ishibuchi et al., 2017). The nadir-point is composed of the maximum objective values found in the non-dominated sets needed to be evaluated. Figure 3 contains an example of a non-dominated solution set and the corresponding nadir-point.

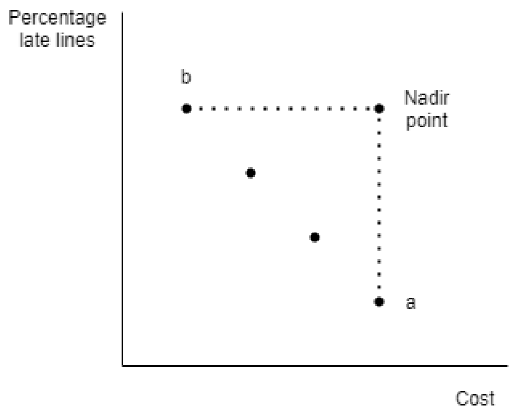


Figure 3: Depiction of a non-dominated solution set with 4 solutions and the corresponding nadir-point. In the figure, a is the most expensive solution and b is the worst solution with regards to the fraction of late lines.

To compare n non-dominated solution sets, a reference point slightly worse than the nadir-point corresponding to the two worst solutions, each regarding one objective, of the n non-dominated solution sets combined. When the reference point lays further from the Pareto front the two extreme solutions (for example solution a and b in Figure 3) will have a bigger contribution to the hyper-volume as opposed to when the reference point is chosen to be close to the nadir point.

The values of the solutions in each non-dominated solution set are scaled to make the data less sensitive with the added benefit of being able to compare hyper-volumes of different instances. This is done by scaling all the solutions relative to the two most extreme solutions found in each of the n non-dominated sets for the same instance. This way the solution with the worst cost solution will have a value of 1 for the cost coordinate and the worst fraction of late lines will have the value of 1 as percentage coordinate. Performing this transformation does result in information loss.

There have not been published papers on comparing different solution sets for the same instance and choosing the reference point for the comparison. After this transformation has taken place it makes sense to pick one reference point that is the same in each case. We have chosen a reference point (1,1).

5.3 Parameter settings

In this research, multiple hyper-parameters are utilized in the various MOACO methods described. In this section, we will explain how the values for all these hyper-parameters influence the algorithms and how the values are set.

To determine which values of the hyper-parameters lead to the best result for each of the algorithms, the training data-set is used. For each version of the algorithm, for each instance, a randomized parameter optimization is employed and their performance is measured by the hyper-volume indicator explained in Section 5.2. The randomized parameter optimization works as follows. For each of the necessary hyper-parameters a random value from the allowed values is selected, leading to a random combination of hyper-parameters. Then, each of the five algorithm versions is run for each of the 283 training instances with this hyper-parameter setting and the hyper-volumes are recorded. This is done 100 times. The hyper-parameter values that lead to the best hyper-volume on average are chosen for the test-set.

Which values can be selected for each hyper-parameter is given in Table 3. The values considered for the hyper-parameters are set based on the hyper-parameter values used in the five papers upon which the five algorithms considered are based.

.

This training resulted in the hyper-parameter settings shown in Table 4. It can be seen that for all algorithms, except CHAC, the same hyper-parameter values were found to produce results with relatively the largest hyper-volume. There was no significant difference in the hyper-volume when 30 or 100 generations were used in each tested instance for each algorithm, yet the run-time increased tremendously, which is why 30 generations were chosen for each of the algorithms.

Hyper parameter	Discrete allowed values
ϕ	{10, 30, 100}
Colony size	{10, 50, 100}
τ_{MAX}	{1, 2, None}
τ_{MIN}	{0.01, 0.1, None}
τ_0	{0.3, 0.5 , 0.7}
α	{1, 2}
β	{4, 0, 3, 1}
θ	{0.01, 0.1, 0.3 }
q_0	{0.05, 0,1, 0.2}

Table 3: Possible settings for the hyper-parameters of the various algorithms.

Models	α	β	θ	m	ϕ	τ_{min}	τ_{max}	q	$\|N_s\|$	S	S'	cycles
g-ACO	1	0	0.01	100	30	0.1	1					
t-ACO	1	0	0.01	100	30	0.1	1	0.05				
CHAC	1	1	0.1	100	30	0.1	1	0.05	4			
na-ACO	1	0	0.01	100	30	0.1	1	0.05				4
c-PACO	1	0	0.01	100	30	0.1	1			50	10	

Table 4: Hyper parameters determined by optimising train data-set

The first hyper-parameter is the colony size m. A large number of ants in the colony means that there are more solutions generated using the same pheromone information, whereas a small number of ants means fewer solutions are generated using the same pheromone information. When there are too many ants in the colony there is a change that many ants generate the same solution, which leads to a less efficient algorithm in terms of both memory use and run-time.

The second hyper-parameter is the number of ant generations ϕ. One generation uses the pheromone trails deposited by the previous generation to choose which item should be supplied by which supplier. A large number of generations could lead to many solutions being found, but if the algorithm converges toward a few good solutions, it might be that the generations later in the algorithm will not find any new solutions anymore. This can also lead to a less efficient algorithm.

The third and fourth hyper-parameters used are τ_{MAX} and τ_{MIN}. Parameter τ_{MIN} is not used for the c-PACO and na-ACO, but for all the others versions they are. Parameter τ_{MAX} is in all models used to initialize the pheromone trails on the arcs of the Supplier Selection problem graphs. In all models, except for c-PACO and na-ACO, they are also used to make sure the pheromone trails do not become too large or too small and thus serve as upper and lower bound. When a pheromone trail of an arc becomes too low, the probability of that arc being chosen will virtually become 0, which could lead to sub-optimal solutions if that the pheromone trails were not in the best solutions, but in good solutions. On the other hand, when the pheromone trails on an arc become too high the probability of that arc being chosen will converge to 1, which could result in the algorithm getting stuck in locally optimal solutions.

Two other hyper-parameters are α and β. These parameters represent the relative importance of the heuristic values and the pheromone trails for the transition decision. A high value of α means that the pheromone trails on the graph have a high impact on the probability the ant will choose a certain arc, whereas a low value of α will mean that this information is not important. A high value of β means that the heuristic values assigned to arcs have a high impact on the probability the ant will choose a certain arc. We can see from Table 4 that β is set to 0 for all algorithms except for CHAC. This means that for the four other algorithms the heuristic values will not be considered in the transition probabilities.

Another hyper-parameter used in each algorithm is θ. This parameter determines the evaporation rate of the pheromone trails. A high evaporation rate (high θ) means that the algorithms value new solutions more than old solutions when updating the pheromone trails, while a lower evaporation rate means the algorithm has a longer memory. Table 4 shows that θ was set to 0.01 for four algorithms. This means that training runs with a long memory resulted in the best non-dominated solution sets.

The hyper-parameter q_0, used in t-ACO, CHAC and na-ACO, determines the importance of exploration versus exploitation. When q_0 is large, exploitation is more important, but will also lead to quicker convergence of the algorithm. With a small q_0, exploration becomes more important, which is the case for the three algorithms that use the pseudo-random proportional transition rule in this research as q_0 is set to 0.05.

Algorithm CHAC requires one extra algorithm-specific hyper-parameter, namely the number of sub-colonies, $|N_s|$. More sub-colonies means fewer ants per sub colony so less exploration of each subspace of the solution space. However, a lower number of sub-colonies means more ants per sub-colony and a more extensive search of fewer sub-spaces. Mora et al. (2009) test four different numbers of sub-colonies, which are 1,4,8 and 16. We find that in our case using 4 colonies results in the best non-dominated solution sets.

Algorithm c-PACO requires two algorithm-specific hyper-parameters, namely S, which is the maximum number of non-dominated solutions allowed in the non-dominated solution set, and S', which is the number of non-dominated solutions which are tested for Pareto efficiency against a newfound solution. A larger number S means the non-dominated solution set is allowed to be bigger and contain more solutions. A larger number of S' means more non-dominated solutions are tested against a new solution in each generation, leading to a higher quality non-dominated solution set, but a longer run-time. To set this we follow the rule used by Angus (2007), who are the developers of the algorithm. They set S and S' equal to $\lfloor\frac{m}{2}\rfloor$ and to $\lfloor\frac{m}{10}\rfloor$ respectively.

Algorithm na-ACO requires an additional hyper-parameter to set the number of cycles the algorithm will make, cycles are made within one generation where the non-dominated solution set is not updated, but pheromone trails are. We set this number of cycles equal to 4. In the research by Afshar et al. (2009) on which the na-ACO has used a ratio of $\frac{1}{25}$ is used to set the number of cycles relative to the number of generations.

5.4 Numerical study

In this section, the results obtained by the experiments are analyzed. Some results are highlighted for illustration purposes and the general results for each of the algorithms are described and compared to one another.

5.4.1 Results related to the test-data set

Each of the algorithms is executed 20 times for each of the test instances, similar to how Mora et al. (2009) produced their results. This is necessary as the algorithms are subject to randomness and running a version multiple times will produce a statistically significant result which will make it possible to rule out the case where one algorithm outperforms the others by chance.

The average hyper-volume metrics obtained over the 20 runs for each algorithm for each instance can be found in Appendix F, Table 11. There were nine instances for which only one solution was part of the non-dominated solution set for each of the algorithm. This resulted in the same hyper-volume for each of the different algorithms. Only finding one solution could mean that there was only one feasible solution or this was the only dominant solution.

There were eleven instances for which the non-dominated solution set obtained by c-PACO provided on average the highest hyper-volumes, two cases in which on average the hyper-volume obtained was equal to the average hyper-volume obtained by na-ACO. In only one instance c-PACO was on average slightly outperformed by na-ACO and CHAC. In almost all of the instances, the standard deviation of the hyper-volume over 20 runs for a specific algorithm was 0.0. This means that the algorithms provide consistent results.

For each of the instances, a figure is provided in Appendix G. These figures contain a mapping of the cost value and fraction of late lines of non-dominated sets for each algorithm for a specific instance. The solution set that is displayed for each algorithm is the one with the best hyper-volume of the 20 runs for a specific instance. The values of the non-dominated solutions are scaled such that the highest cost found in all runs is set to 1 and the total cost of all other non-dominated solutions are scaled relative to the highest cost solution. The same transformation is made to the fraction of late lines. This scaling done to make the data less sensitive. The figures also contain data on the actual percentage of late-lines and the actual total cost of each of the instances. These values are scaled to the same scale as the solution values in the non-dominated solution set. The figures in Appendix G give a clear view of the structure of the non-dominated solution sets for each algorithm. It shows that the c-PACO algorithm non-dominated solutions, in many instances, dominate the non-dominated solution sets provided by other algorithms.

The average run-time of each algorithm for each instance can be found in Appendix E, Table 10. The run-time is measured from the moment slightly after the data is imported until the moment the algorithm terminates. It is clear that for each instance the c-PACO algorithm obtained the fasted run-time, while na-ACO has the longest run-time on average. The run-time of the na-ACO algorithm is on average, measured over each run and each instance, twice as long. The difference is really visible in instances 2,13 and 26 where the na-ACO runtime is on average over a minute while for the c-PACO this is around 20 seconds. Incidentally these are the instances with the highest complexity as can be seen in Appendix C, Table 8.

The run-times are less than half a minute for each of the algorithms, except na-ACO in some instances, which is also displayed in Figure 4. This is a good result as currently decision-makers are taking anywhere from half an hour to multiple hours on one order. Using any of the algorithms used in this research will therefore result in a significant decrease in time spent on making orders.

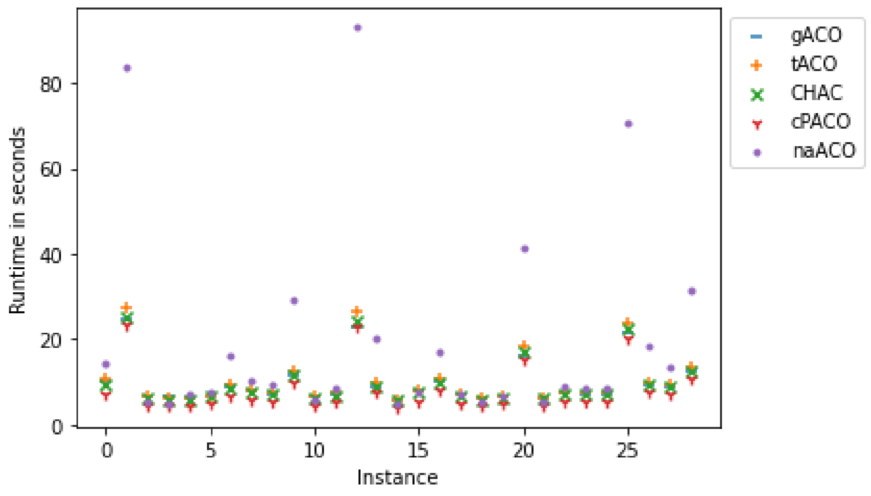


Figure 4: Average run-time in seconds of each algorithm for each instance.

The c-PACO algorithm is on average the fastest algorithm in each of the instances, which can be seen in Figure 4. This can be explained by the fact that in each generation a new solution is not compared with the entire non-dominated set, but only with a subset of the non-dominated solutions. Besides, the pheromone updating is done via the NSGA-II algorithm which is a very fast algorithm. The na-ACO algorithm is on average slower than the other algorithms, which is due to the extra cycles within one generation of the algorithm. This makes the na-ACO algorithm more computationally expensive. The other three algorithms, g-ACO, t-ACO and CHAC, have very similar run times.

Figure 5 shows the (best of 20 runs) non-dominated solution set obtained by the algorithms for Instance 27. This is the same figure as can be found in Appendix G, except for that we omitted the actual costs for clarity of the figure. We have chosen this instance to show how the hyper-volume metric behaves. The reference point is, as explained in Section 5.2 set to (1,1). The hyper-volume for each of the algorithms for Instance 27, except for t-ACO, is 0.167 even though thec-PACO non-dominated solution set dominates the other sets. It can be seen from the figure that the three solutions in the non-dominated set ofc-PACO have lower cost and a lower fraction of late lines than non-dominated solutions of other algorithms in the same region of the solution space. The reason that the hyper-volume value is the same is that the other algorithms have a non-dominated solution set with more extreme solutions, a more even spread and more solutions, which are three other qualities desired in a non-dominated set besides closeness to the actual Pareto Front. This illustrates how the hyper-volume works in practice and shows clearly how the measure takes into account various desired qualities a non-dominated solution set should have.

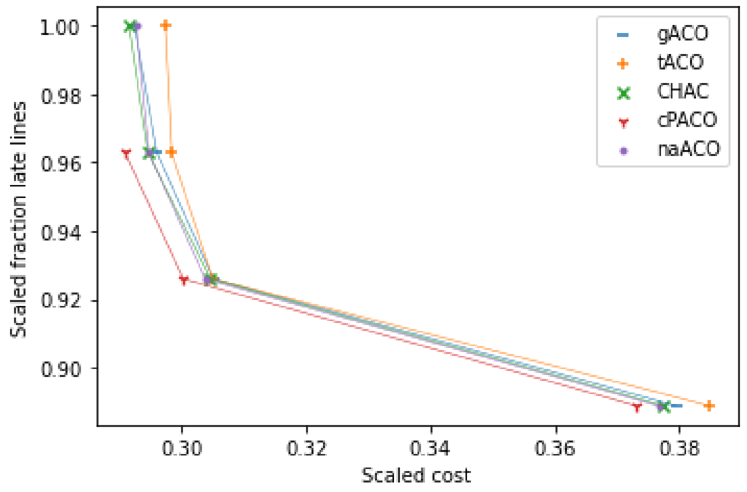
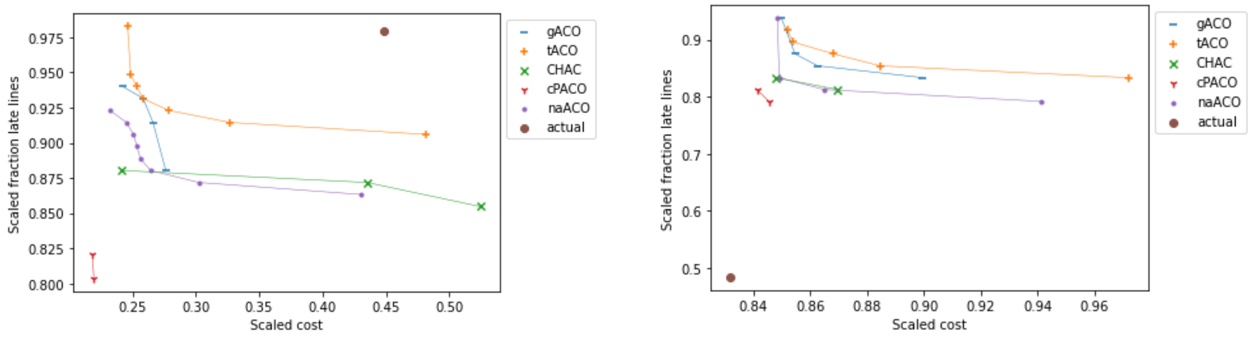


Figure 5: Instance 27

For other instances, such as Instance 2 and 10 displayed in Figure 6a and 6b, c-PACO has the lowest number of non-dominated, yet because the non-dominated solution set dominates the other sets by so much, the hyper-volume value is much higher than the others in both cases. Instance 2 and 10 have an average hyper-volume for the c-PACO of 0.251 and 0.08 respectively. These and the hyper-volume for all other algorithms is displayed in Table 5, which is small version of Table 11 in Appendix F. This indicates that the c-PACO algorithm is able to find solutions that the other algorithms are unable to. Which makes the c-PACO the better algorithm version. This also again shows the behaviour of the hyper-volume measure.

Instance	g-ACO	t-ACO	CHAC	c-PACO	na-ACO
2	0.18 (0.01)	0.17 (0.01)	0.19 (0.01)	0.25 (0.01)	0.20 (0.01)
10	0.07 (0.0)	0.06 (0.0)	0.07 (0.0)	0.08 (0.0)	0.08 (0.0)

Table 5: Average hyper-volume results of 20 test-runs for Instance 2 and 10 for each algorithm version. The standard deviation is the number in brackets.



(a) Instance 2

(b) Instance 10

Figure 6

The g-ACO was expected to have most solutions in the extreme parts of the solution space, yet in almost all instances this is not the case. When considering the run-time of the g-ACO algorithm, it is very similar to the run-times of both t-ACO and CHAC. When considering the average hyper-volume for each instance, as can be found in Appendix F, Table 11, we can see that g-ACO has similar or slightly better hyper-volumes than t-ACO and similar or slightly worse than CHAC.

The t-ACO algorithm has two distinctly different elements than the g-ACO, namely weighted preferences for the ants and the pseudo-random proportional transition procedure. Those two elements were expected to result in non-dominated solution sets with better spread compared to g-ACO and a better balance between exploration and exploitation and were expected to find a better non-dominated solution set than g-ACO, but considering the average hyper-volumes this was not the case in practice. This unexpected result has multiple explanations. It is possible that the hyper-parameters where not suitable after all, making the algorithm prone to exploitation rather than exploration which we desired from this algorithm. It could also be that the algorithm converges early in the process.

The CHAC algorithm introduced the island method, where multiple ants had the same preferences and only shared information with those ants. This structure was expected to provide a faster algorithm and a better exploration of the solution space. The run-time of the CHAC however, does not differ from the speed of the g-ACO significantly. Another way to deal with this is to run the different sub-colonies in parallel. For 10 out of 29 instances the average hyper-volume of the CHAC non-dominated sets are larger than those of the g-ACO and in Instance 23 CHAC has the largest hyper-volume, alongside na-ACO, of all algorithms. This makes the CHAC a suitable version of the algorihtm to use in practice.

The c-PACO algorithm introduced a new type of solutions storage as well as Pareto testing. Both these innovations were expected to make the algorithm more efficient in terms of memory use. This definitely was the case in practice as for each of the instances, for each of the runs the c-PACO algorithm has the shortest run times. In 9 of 29 instances the c-PACO non-dominated solution set with the highest hyper-volume over all 20 runs, strictly dominates all other non-dominated solution sets obtained for those instances. In 13 instances the c-PACO algorithm obtained, on average, the biggest hyper-volume. Between the short run-time and the ability to find non-dominated solutions better than the other algorithms, we conclude that c-PACO is the best algorithm to use in practice. A drawback of the algorithm is that in the experiments we often see that the non-dominated solution set consists of few solutions. When a solution set contains few solutions this means that the decision-makers are less flexible when choosing a solution that suits the needs for their mission specifically.

The na-ACO algorithm introduced the non-dominated solution archive which was supposed to make the algorithm efficient. Besides this, it also has two ant colonies that evaluate each other's solutions. This was expected to help the algorithm find solutions that are spread out over the non-dominated solution set. The na-ACO algorithm however is much slower than all algorithms, on average even twice as slow, yet it does find non-dominated solution sets which are similar to the c-PACO algorithm, only with slightly worse hyper-volumes.

In this section we analyzed the results obtained from all the algorithms for various instances. We determined how well the algorithms performed using historical data based on run-time and hyper-volume. We find that each of the algorithms have very short run-times. We also found that the c-PACO generally outperforms all other algorithms in terms of run-time and hyper-volume. We have not yet compared the results of the algorithms with the historical purchase orders. This is provided in the next section.

5.4.2 Comparison to actual purchases

In this section, the actual costs for each instance are compared to the results obtained in this research. Appendix C, Table 9 contains a scaled overview of the costs and fraction of late lines, scaled with the same values as the non-dominated solution sets. In this research only the cost objective is evaluated, as in practice the lead-times of the purchase orders were not recorded consistently. We did use lead-times estimates in this research, as those were based on data recorded over a longer period. Taking only one data point to evaluate our results will not result in valid conclusions.

To evaluate the percentage cost difference between the actual total cost and the costs obtained by the different methods for each instance, we compare three costs per method with the actual cost of the instance. Namely, the non-dominated solution with the highest cost obtained in the 20 runs, the non-dominated solution with the lowest cost obtained in the 20 runs and the average cost of all non-dominated solutions found in the 20 runs. The percentage value is obtained by $(1 - \frac{\text{Cost found by algorithm}}{\text{Actual cost of total purchase}}) * 100$.

Table 6 contains a summary of the results obtained in this section. It displays the averages over all instances for each of the five methods. Each of the methods results on average in an expected cost decrease of at-least 5%. When the worst-case cost, or maximum cost, is considered there is on average an expected cost increase in all of the methods, while only considering the best-case costs for all instances leads to an average cost decrease of at-least 15%.

Instance	Maximum	Minimum	Average
g-ACO	11.97%	-16.10%	-6.64%
t-ACO	16.24%	-15.96%	-5.88%
CHAC	10.23%	-16.36%	-7.30%
c-PACO	2.33%	-17.10%	-9.88%
na-ACO	12.60%	-16.22%	-7.36%

Table 6: This table contains the average over all instances per algorithm version for the maximum, minimum and average percentage cost difference compared to the actual costs for each instance.

Table 7 contains the comparison results for each of the instances for the c-PACO method. The detailed results for all other methods can be found in Appendix H. From the previous section it can be concluded that c-PACO performs best in terms of efficiency, run-time and quality of the non-dominated solution sets obtained, which is why this algorithm will be proposed to be used in the tool. For conciseness this is the only algorithm that will be analysed in this section.

	Maximum		Minimum		Average	
Instance	Cost	Percentage difference	Cost	Percentage difference	Cost	Percentage difference
1	1.00	-5.76%	1.00	-5.76%	1.00	-5.76%
2	0.23	-48.57%	0.22	-51.93%	0.22	-50.23%
3	1.00	0.00%	1.00	0.00%	1.00	0.00%
4	0.38	0.44%	0.38	-0.35%	0.38	0.05%%
5	1.00	-23.41%	1.00	-23.41%	1.00	-23.41%
6	1.00	-52.89%	1.00	-52.89 %	1.00	-52.89%
7	1.00	37.58%	0.83	13.92%	0.88	21.01%
8	0.95	-34.98%	0.95	-34.98%	0.95	-34.98%
9	0.99	7.90%	0.99	7.90%	0.99	7.90%
10	0.85	1.75%	0.84	1.04%	0.84	1.34%
11	1.00	-14.08%	1.00	-14.08%	1.00	-14.08%
12	1.00	0.19%	1.00	0.19%	1.00	0.19%
13	0.91	-9.00%	0.90	-9.81%	0.9‘	-9.39%
14	1.00	27.28%	0.77	-1.79%	0.80	2.17%
15	1.00	-94.83%	1.00	-94.83%	1.00	-94.83%
16	1.00	138.20%	0.74	76.36%	0.87	107.28%
17	0.69	67.01%	0.25	-40.51%	0.47	13.25%
18	1.00	-47.98%	0.96	-49.92%	0.981	-48.95%
19	1.00	17.34%	1.00	17.34%	1.00	17.34%
20	1.00	40.41%	1.00	40.41%	1.00	40.41%
21	0.72	-21.36%	0.61	-33.87%	0.65	-29.25%
22	1.00	-13.69%	1.00	-13.69%	1.00	-13.69%
23	0.90	130.26%	0.33	-15.08%	0.57	46.47%
24	1.00	22.80%	0.31	-62.57%	0.53	-34.60%
25	1.00	-7.54%	0.79	-27.33%	0.89	-17.43%
26	0.53	-12.91%	0.52	-14.29%	0.52	-13.78%
27	1.00	-3.27%	0.29	-71.8%7	0.34	-67.13%
28	1.00	42.51%	1.00	42.51%	1.00	42.51%
29	0.36	-75.94	0.35	-76.53%	0.35	-76.19%
Average	0.88	2.33%	0.76	-17.10%	0.80	-9.89%

Table 7: This table contains the maximum, minimum and average expected cost obtained by c-PACO for each instance . It also contains the percentage difference between the maximum, minimum and average cost compared to the actual cost.

The range of percentage difference that can be seen in Table 7 is very wide, ranging from an expected cost decrease of 95 % in instance 15 to an expected cost increase of 138 % in the highest cost case of instance 16. The major expected cost increase in the worst-case could be explained in the two following ways. Firstly, it is possible that in some cases we vastly overestimate the AFAM costs that are associated with a possible solution. For instance, when no AFAM costs were recorded for certain suppliers an AFAM cost of 76 euro was assumed, see Section 5.1.2, while it could also be possible that no record of AFAM costs meant that no AFAM costs need to be paid for that supplier. Eventhough the assumption of positive AFAM costs was made in this research, it could be that AFAM costs are already integrated in the pricing agreements. This could lead to situations in which the actual total cost is lower than the best case expected cost obtained by c-PACO such as in Instance 16 and 20, especially if the total actual cost was low. This problem will solve itself in the long run, as more data means the estimates will get more reliable. Secondly, it is possible that the high costs are incurred in solutions where to total percentage of late-lines is low. This is the case in instances 17 and 23, where the highest cost would mean a significant cost increase of 67 and 130 percent respectively, yet the lowest cost leads to a cost decrease of 40 and 15 percent respectively. This means that the solution with worst-case cost is not realistically the cost that will be incurred in practice. This is not a surprising result because this is in line with the structure of the problem.

There is also a difference in magnitude of the increase or decrease, ranging from a percentage difference of less than a percent to a difference of 138 percent. This is due to the fact that some instances consist of many lines that can lead to very expensive solutions where a difference of some euro's gives a small percentage difference, while other instances consist of few lines, where a difference of some euro's lead to very different results and thus a high percentage difference. This means we cannot compare the percentage differences for different instances directly.

On average the proposed c-PACO method will lead to a significant expected cost decrease and in practice the cost decrease might even be bigger than is expected in this research. This means that the decision tool provides added value in threefold. It leads to an expected cost decrease on average, the time spent on making the item assignments is decreased dramatically and the tool provides better insight into the structure of the item assignment problem for the decision makers.

6 Discussion and conclusion

In this research, we investigated the Supplier Selection problem specific to Médicines sans Frontières. The Supplier Selection problem in this research consists of selecting which item is purchased at which supplier, where the suppliers could have a minimum order value of quantity. The problem has two objectives, namely minimizing the total cost of an item assignment and minimizing the lead-time. The transport, service and other costs, together called AFAM costs, are uncertain as is the lead-time. The aim of the research is to find one or more item assignments that are non-dominated by each other with respect to lead-time and total cost. These item assignments are presented to the decision-makers who choose which of the objectives is more relevant.

The item assignments were formerly made by decision-makers that had an assortment of different tasks, of which the item assignments was one. The creation of each item assignment could take anywhere from 30 minutes to multiple hours. They aspired to find a feasible solution and to obtain this they started with selecting the supplier with the lowest per piece price for an item. The decision-makers had no insight into the actual AFAM costs and actual lead times, which complicated decision making. The goal of this research is to create a decision tool that provides the decision-makers with non-dominated solutions to the item assignment within a reasonable time, while simultaneously giving them insight in the structure of the problem. A secondary goal is to find item assignments that will in the long-run result in an improvement in the KPI's, either by reducing total costs or lead-time.

As Médicines sans Frontières is a humanitarian organization they have a rare supply chain type, as the organization has as the ultimate goal to provide people with adequate health care. This is very different from the supply chains typically investigated in existing literature, which have as the ultimate goal to pursue profits. Therefore, the Supplier Selection problem has not been investigated in the context of the humanitarian supply chain, which is one of the main contributions of my research.

To find a non-dominated solution set, artificial intelligence has taken a flight in popularity. It has been proposed that the Supplier Selection problem with multiple objectives can be solved with the Multi-objective ant colony optimization (MOACO) algorithm, which is a framework with many different proposed versions. Using the MOACO framework to solve the Supplier Selection problem with simulated data, yet has never been tested in practice. This is another contribution to existing literature, as we implemented multiple versions of the MOACO algorithm and used real-life data provided by Médicines sans Frontières.

In this research, five different MOACO versions are investigated. Two of those are basic versions which have been shown to be a good benchmark and three are state-of-the-art versions which each incorporate innovative structures to overcome disadvantages of the basic versions of the MOACO, such as fast convergence to local optima.

Each of the five algorithms proposed, when tested with our test data, resulted in high-quality solutions with a run-time ranging from seconds to two minutes for the most complex instances. One algorithm outperformed the other both in speed and solution quality. This is the *Crowding population-based Ant Colony Optimization* as proposed by Angus (2007) and adapted to the problem at hand. This is the method we propose to use in the decision tool.

We found that the tool will on average, over the instances we tested, be able to result in at least a 9.89 % cost decrease. If the decision-makers always choose the solution with the lowest cost out of all solutions, the tool will help to achieve on average a 17.10 % cost decrease. The available procurement budget that will be influenced by the introduction of the decision tool is in the most conservative case 20 million euros per year, in which case the tool will save MSF on average more than 1.98 million euros per year. If the most cost-efficient solution is selected by the decision-makers in each case, the organisation could decrease their procurement cost by around 3.42 million euros per year.

The proposed solution to the Supplier Selection problem provides added value to the status quo of Médicines sans Frontières in the following ways. Firstly, the tool created with the algorithm provides a set of solutions in under a minute, simplifying and shortening the procurement process tremendously, which also reduces the amount of labour needed for procurement. Secondly, the results found in this research suggest that on average it is possible to find a significant expected cost reduction. Lastly, the solutions provided give a good insight into the structure of the problem to the decision-makers.

One limitation of this research is the limited amount of available historical purchases for each possible item. This resulted in estimation of lead-time and AFAM costs which can diverge from the actual lead-time and AFAM costs, which results in less robust solutions. This problem, however, will become less prominent as more data becomes available every day. In future research, other methods for lead-time and AFAM costs can be explored to test the accuracy of the predictions and eventually find more accurate predictions.

Another limitation of this research is that we used the same hyper-parameter settings for each instance, while it might differ depending on the structure of the instance which hyper-parameter settings result in better non-dominated solution settings. In future research meta-learned hyper-parameter optimization can be investigated to avoid this problem.

This research considered two objectives, while in practice more objectives are relevant. For example, the remaining shelf-life of products such as medicines or the quality of the items. In further research adding more objectives to the problem to be able to include more information in the decision making process can be explored.

References

Abdollahzadeh, H., & Atashgar, K. (2017). Optimal design of a multi-state system with uncertainty in supplier selection. *Computers & Industrial Engineering*, *105*, 411–424.

Afshar, A., Sharifi, F., & Jalali, M. R. (2009, 4). Non-dominated archiving multi-colony ant algorithm for multi-objective optimization: Application to multi-purpose reservoir operation. *Engineering Optimization*, *41*, 313-325.

Alaya, I., Solnon, C., & Ghédira, K. (2007). Ant colony optimization for multi-objective optimization problems. *19th IEEE international conference on tools with artificial intelligence (ICTAI 2007)*, 450-457.

Andersen, K. A., Jörnsten, K., & Lind, M. (1996). On bicriterion minimal spanning trees: an approximation. *Computers & Operations Research*, *23*(12), 1171–1182.

Angus, D. (2007). Crowding population-based ant colony optimisation for the multi-objective travelling salesman problem. *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Multicriteria Decision Making, MCDM 2007*, 333-340.

Angus, D., & Woodward, C. (2009). Multiple objective ant colony optimisation. *Swarm Intelligence*, *3*(1), 69–85.

Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., & Salomon, L. (2020). Performance indicators in multi-objective optimization. *European Journal of Operational Research*, *292*(2), 397–422.

Bartle, R. G. (2014). *The elements of integration and lebesgue measure.* John Wiley & Sons.

Bates, B. R. (2005). Hope in hell: Inside the world of doctors without borders. *Journal of the National Medical Association*, *97*(11), 1575.

Burkard, R., Dell'Amico, M., & Martello, S. (2012). *Assignment problems: revised reprint.* SIAM.

Chai, J., Liu, J. N., & Ngai, E. W. (2013). Application of decision-making techniques in supplier selection: A systematic review of literature. *Expert Systems with Applications*, *40*(10), 3872–3885.

Chai, J., & Ngai, E. W. (2020). Decision-making techniques in supplier selection: Recent accomplishments and what lies ahead. *Expert Systems with Applications*, *140*, 112903.

Chopra, S., & Meindl, P. (2007). Supply chain management. strategy, planning & operation. In *Das summa summarum des management* (pp. 265–275). Springer.

Coello, C. A. C., Lamont, G. B., Van Veldhuizen, D. A., et al. (2007). *Evolutionary algorithms for solving multi-objective problems* (Vol. 5). Springer.

Deb, K. (2014). Multi-objective optimization. In *Search methodologies* (pp. 403–449). Springer.

Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *International conference on parallel problem solving from nature*, 849–858.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Deneubourg, J.-L., & Goss, S. (1989). Collective patterns and decision-making. *Ethology Ecology & Evolution*, *1*(4), 295–311.

Dorigo, M., & Gambardella, L. M. (1997). Ant colonies for the travelling salesman problem. *Biosystems*, *43*(2), 73–81.

Falcón-Cardona, J. G., Leguizamón, G., Coello, C. A. C., & Tapia, M. G. C. (2020). Multi-objective ant colony optimization: An updated taxonomy and review of approaches. *Preprint available at www. researchgate. net*.

García-Martínez, C., Cordón, O., & Herrera, F. (2007). A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. *European journal of operational research*, *180*(1), 116–148.

Ghodsypour, S. H., & O'Brien, C. (1998). A decision support system for supplier selection using an integrated analytic hierarchy process and linear programming. *International Journal of Production Economics*, *56*, 199–212.

Guntsch, M., & Middendorf, M. (2002). Applying population based ACO to dynamic optimization problems. , 111–122.

Gutjahr, W. J. (2004). S-ACO: An ant-based approach to combinatorial optimization under uncertainty. , 238–249.

Iredi, S., Merkle, D., & Middendorf, M. (2001). Bi-criterion optimization with multi colony ant algorithms. , 359–372.

Ishibuchi, H., Imada, R., Setoguchi, Y., & Nojima, Y. (2017). Reference point specification in hypervolume calculation for fair comparison and efficient search. , 585–592.

Jaszkiewicz, A. (2004). Evaluation of multiple objective metaheuristics. In *Metaheuristics for multiobjective optimisation* (pp. 65–89). Springer.

Knowles, J. D., Thiele, L., & Zitzler, E. (2006). A tutorial on the performance assessment of stochastic multiobjective optimizers. *TIK-report*, *214*.

Krumke, S. O., & Thielen, C. (2013). The generalized assignment problem with minimum quantities. *European Journal of Operational Research*, *228*(1), 46–55.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, *2*(1-2), 83–97.

Lambert, D. M., & Cooper, M. C. (2000). Issues in supply chain management. *Industrial Marketing Management*, *29*(1), 65–83.

Luan, J., Yao, Z., Zhao, F., & Song, X. (2019). A novel method to solve supplier selection problem: Hybrid algorithm of genetic algorithm and ant colony optimization. *Mathematics and Computers in Simulation*, *156*, 294–309.

Mishra, K., & Harit, S. (2010). A fast algorithm for finding the non dominated set in multi objective optimization. *International Journal of Computer Applications*, *1*(25), 35–39.

Mora, A. M., García-Sánchez, P., Merelo, J. J., & Castillo, P. A. (2013). Migration study on a pareto-based island model for MOACOs. *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference*, 57-64. doi: 10.1145/2463372.2463390

Mora, A. M., Merelo, J. J., Laredo, J. L. J., Millán, C., & Torrecillas, J. (2009). Chac, a MOACO algorithm for computation of bi-criteria military unit path in the battlefield: Presentation and first results. *International Journal of Intelligent Systems*, *24*(7), 818–843.

MSF. (2021). Apu dataset vendor selection.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, *5*(1), 32–38.

Qu, B.-Y., Zhu, Y., Jiao, Y., Wu, M., Suganthan, P. N., & Liang, J. (2018). A survey on multi-objective evolutionary algorithms for the solution of the environmental/economic dispatch problems. *Swarm and Evolutionary Computation*, *38*, 1–11.

Rizk, C., & Arnaout, J.-P. (2012). ACO for the surgical cases assignment problem. *Journal of Medical Systems*, *36*(3), 1891–1899.

Sonmez, M. (2006). *Review and critique of supplier selection process and practices.* Citeseer.

Stützle, T., & Dorigo, M. (1999). ACO algorithms for the quadratic assignment problem. *New ideas in optimization*(C50), 33.

Tomasini, R., Van Wassenhove, L., & Van Wassenhove, L. (2009). *Humanitarian logistics.* Springer.

Tomasini, R. M., & Van Wassenhove, L. N. (2009). From preparedness to partnerships: case study research on humanitarian logistics. *International Transactions in Operational Research*, *16*(5), 549–559.

Weber, C. A., Current, J. R., & Benton, W. (1991). Vendor selection criteria and methods. *European Journal of Operational Research*, *50*(1), 2–18.

Yang, Y., Wu, G., Chen, J., & Dai, W. (2010, 3). Multi-objective optimization based on ant colony optimization in grid over optical burst switching networks. *Expert Systems with Applications*, *37*, 1769-1775.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE transactions on Evolutionary Computation*, *3*(4), 257–271.

A Flow representation of order sequence

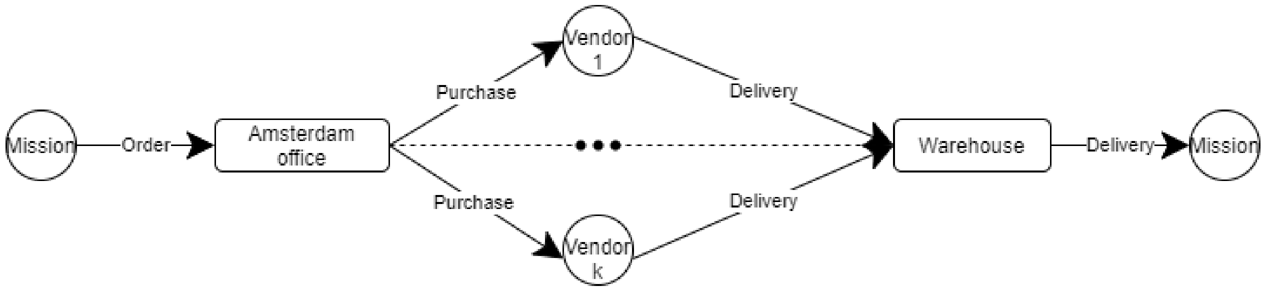


Figure 7: Flow representation of the steps necessary to buy items.

B Flowcharts different MOACO methods

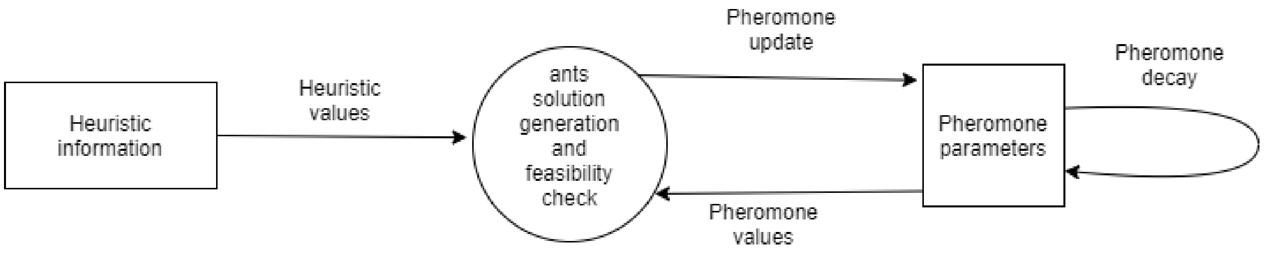


Figure 8: Sequence of events as used by the g-ACO and t-ACO algorithms. Note, this figure is adapted from the work of Angus & Woodward (2009).

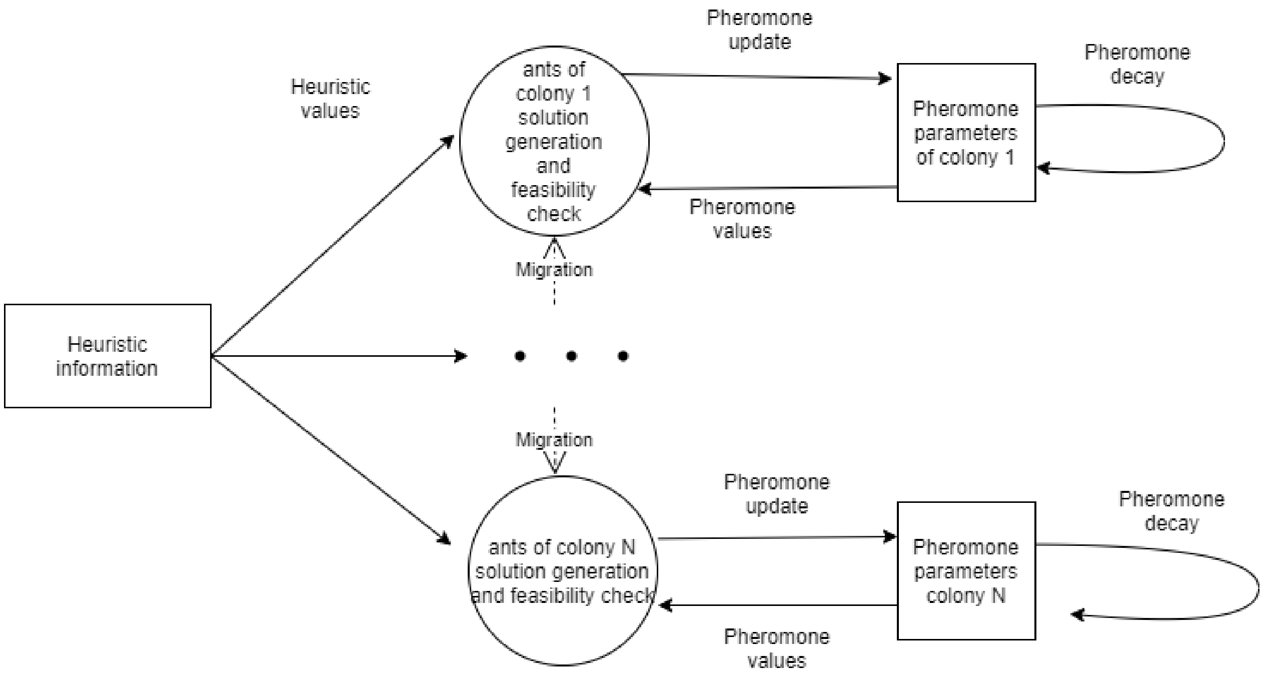

Heuristic values
ants of colony 1 solution generation and feasibility check
Pheromone update
Pheromone parameters of colony 1
Pheromone decay
Pheromone values
Migration
Heuristic information
Migration
ants of colony N solution generation and feasibility check
Pheromone update
Pheromone parameters colony N
Pheromone decay
Pheromone values

Figure 9: Sequence of events as used by the CHAC algorithm.

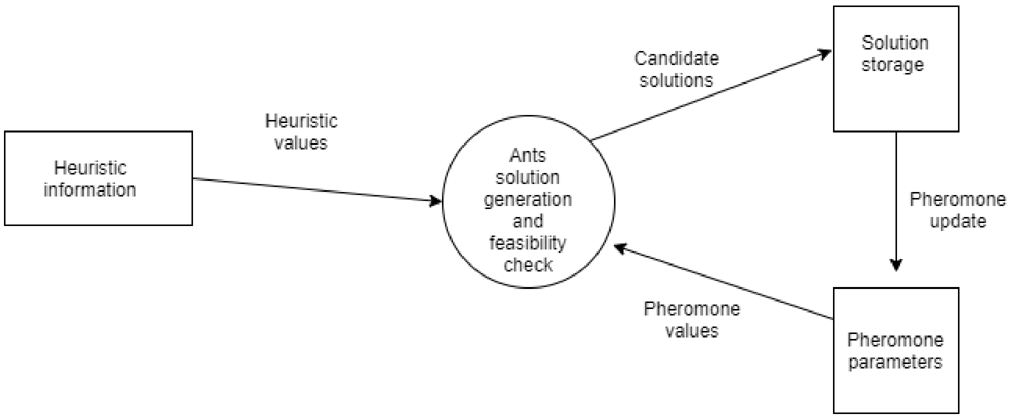


Figure 10: Sequence of events as used by the c-PACO algorithm.

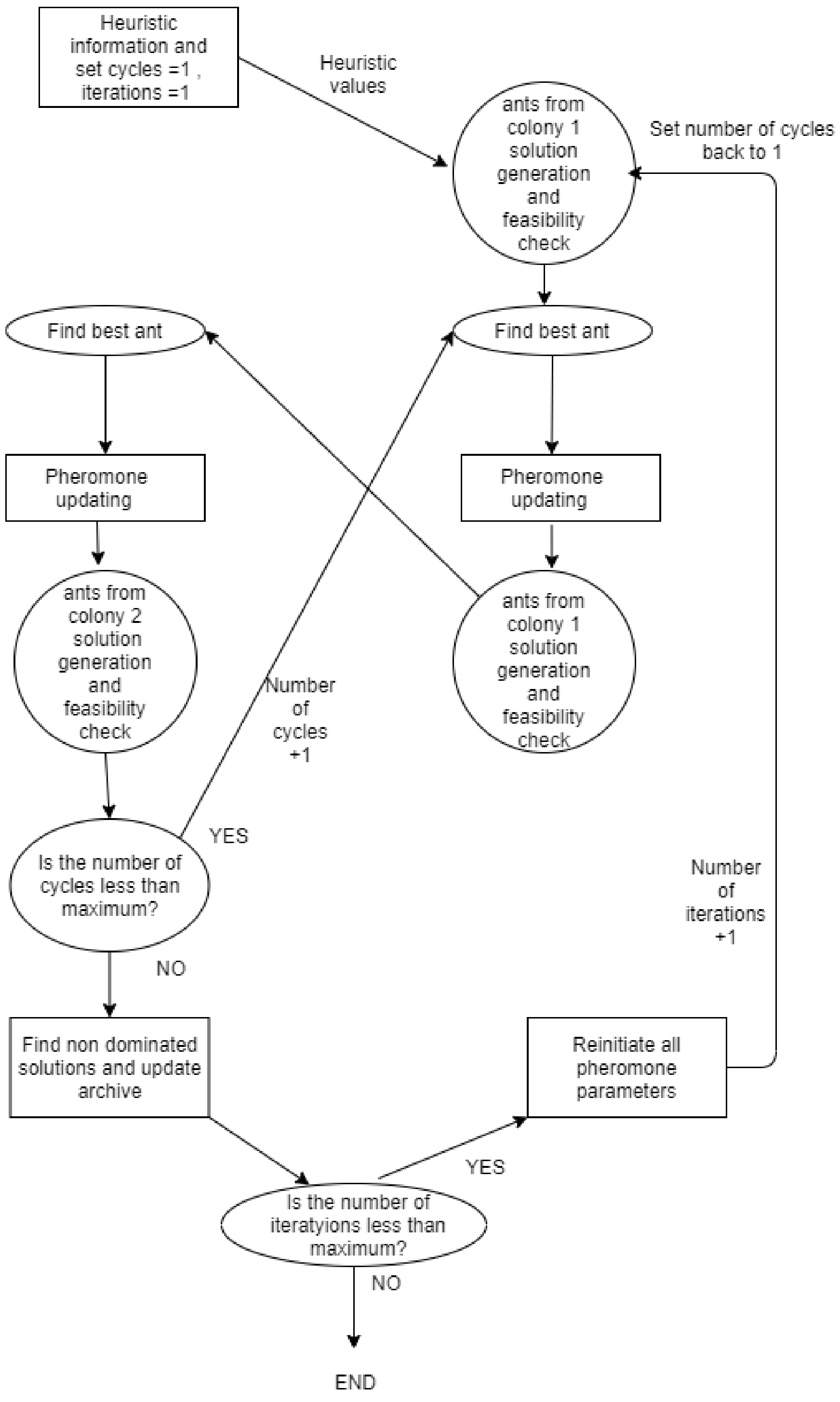

Heuristic information and set cycles =1 , iterations =1
Heuristic values
ants from colony 1 solution generation and feasibility check
Set number of cycles back to 1
Find best ant
Find best ant
Pheromone updating
Pheromone updating
ants from colony 2 solution generation and feasibility check
ants from colony 1 solution generation and feasibility check
Number of cycles +1
YES
Is the number of cycles less than maximum?
NO
Number of iterations +1
Find non dominated solutions and update archive
Reinitiate all pheromone parameters
YES
Is the number of iteratyions less than maximum?
NO
END

Figure 11: Sequence of events as used by the na-ACO algorithm.

C Details test instances

In this section a Table 8 is provided. The table contains details on each of the 29 test-instances, such as the number of lines in the order.

Instance	number of lines	complexity	number of possible suppliers	average number of suppliers per line
1	16	24	10	1.3125
2	123	6.41E+60	45	3.430894
3	3	2	2	1.333333
4	2	2	2	1.5
5	9	48	7	2.111111
6	8	432	6	2.375
7	24	172800	16	1.916667
8	10	41472	14	3.4
9	11	180	7	2.272727
10	50	2.55E+09	20	2.2
11	3	8	4	3.666667
12	8	144	4	2.375
13	226	5.88E+56	6	1.840708
14	44	49152	14	1.386364
15	2	12	4	3.5
16	3	20	5	4
17	16	2.62E+08	14	4.75
18	6	8	2	2.166667
19	2	3	3	2
20	4	24	3	3
21	39	3.65E+31	29	7.435897
22	3	4	2	2
23	7	6000	8	4.428571
24	8	1296	7	2.875
25	6	64	4	3.833333
26	114	2.36E+49	63	2.982456
27	27	9.56E+08	20	2.481481
28	19	72000	8	2.210526
29	49	3.51E+19	30	2.836735

Table 8: Details about each of the test instances

D Historical objective values of purchase orders

This section contains Table 9. The table contains the actual scaled objective values for each of the test-instances.

	Scaled actual cost	Scaled actual percentage late lines
1	1.061168	0.578462
2	0.448012	0.979432
3	1	0.33
4	0.381446	0.5
5	1.305686	1
6	2.122365	0.5
7	0.726866	0.718751
8	1.457917	0
9	0.91252	1
10	0.832135	0.483631
11	1.16388	1
12	0.998066	0.888889
13	0.999581	0.987234
14	0.785675	0.837747
15	19.34824	0.666667
16	0.41981	0.750751
17	0.411819	1
18	1.922454	0.40016
19	0.852246	1
20	0.712219	1.333333
21	0.918259	1
22	1.158665	1
23	0.389146	1
24	0.814356	1
25	1.081522	1
26	0.605273	0.924983
27	1.033793	1
28	0.701703	1
29	1.477672	0.471192

Table 9: Table contains the actual values of the orders scaled to the maximum values found with our algorithms.

E Run-time test instances

This section contains Table 10. The table contains the average and standard deviation run-time over the 20 experiments for each instance and each algorithm version.

Instance	gACO	tACO	CHAC	cPACO	naACO
1	10.41 (2.69)	10.77 (2.24)	9.45 (2.25)	**7.01 (1.26)**	14.18 (2.5)
2	24.56 (5.76)	27.48 (5.32)	25.33 (4.79)	**23.4 (4.88)**	83.63 (14.54)
3	6.49 (0.29)	6.48 (0.25)	5.99 (0.22)	**4.25 (0.16)**	5.23 (0.21)
4	6.43 (0.39)	6.37 (0.21)	5.87 (0.25)	**4.18 (0.2)**	4.9 (0.3)
5	6.19 (1.11)	6.19 (1.23)	5.75 (1.14)	**4.35 (0.84)**	6.97 (1.27)
6	6.61 (0.95)	6.81 (0.99)	6.43 (0.72)	**4.61 (0.57)**	7.38 (0.91)
7	8.91 (0.26)	9.35 (0.35)	8.6 (0.27)	**6.81 (0.19)**	15.88 (0.32)
8	7.8 (0.29)	8.05 (0.27)	7.45 (0.27)	**5.66 (0.24)**	10.36 (0.24)
9	7.28 (0.26)	7.49 (0.27)	7.02 (0.22)	**5.34 (0.21)**	9.41 (0.26)
10	11.39 (0.3)	12.42 (0.48)	11.61 (0.33)	**9.58 (0.21)**	29.06 (0.53)
11	6.64 (0.23)	6.56 (0.17)	6.23 (0.36)	**4.5 (0.18)**	5.73 (0.2)
12	7.14 (0.21)	7.35 (0.25)	6.81 (0.24)	**5.17 (0.22)**	8.22 (0.25)
13	22.93 (3.17)	26.33 (3.41)	24.28 (3.37)	**22.72 (2.85)**	93.1 (10.43)
14	8.99 (2.25)	9.98 (3.63)	8.98 (2.44)	**7.56 (2.01)**	20.24 (5.13)
15	5.87 (1.75)	5.94 (1.81)	5.68 (1.66)	**3.96 (1.09)**	4.76 (1.31)
16	7.94 (0.29)	8.17 (0.29)	7.47 (0.21)	**5.39 (0.22)**	7.34 (0.33)
17	10.03 (0.69)	10.77 (0.67)	9.89 (0.62)	**8.08 (0.6)**	17.03 (1.02)
18	6.9 (0.29)	7.07 (0.24)	6.64 (0.22)	**4.88 (0.25)**	7.22 (0.28)
19	6.22 (0.22)	6.32 (0.23)	5.83 (0.21)	**4.24 (0.2)**	5.08 (0.25)
20	6.78 (0.39)	6.8 (0.34)	6.37 (0.37)	**4.69 (0.24)**	6.55 (0.23)
21	16.13 (0.32)	18.35 (0.45)	17.02 (0.34)	**14.98 (0.33)**	41.44 (0.88)
22	6.37 (0.29)	6.35 (0.29)	5.99 (0.29)	**4.35 (0.26)**	5.28 (0.22)
23	7.31 (0.23)	7.62 (0.35)	6.97 (0.26)	**5.31 (0.32)**	8.9 (0.3)
24	7.25 (0.23)	7.44 (0.26)	6.98 (0.3)	**5.21 (0.21)**	8.42 (0.28)
25	7.13 (0.23)	7.56 (0.32)	6.92 (0.26)	**5.16 (0.24)**	8.55 (0.24)
26	21.29 (0.5)	23.93 (0.44)	22.4 (0.36)	**20.37 (0.33)**	70.69 (0.51)
27	9.42 (0.45)	9.96 (0.28)	9.29 (0.26)	**7.45 (0.33)**	18.3 (0.32)
28	8.9 (0.31)	9.37 (0.32)	8.71 (0.33)	**6.92 (0.27)**	13.44 (0.32)
29	12.35 (0.33)	13.41 (0.39)	12.47 (0.22)	**10.84 (0.41)**	31.46 (0.29)

Table 10: Average run-time in seconds of 20 test-runs per test-instance per model. The standard deviation is the number in brackets. The average runt-time of the model that are the best for a given instance are made bold.

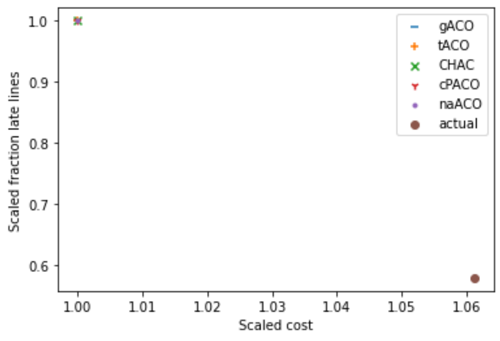
F Hyper-volume test instances

This section contains Table 11. The table contains the average and standard deviation hyper-volume over the 20 experiments for each instance and each algorithm version.

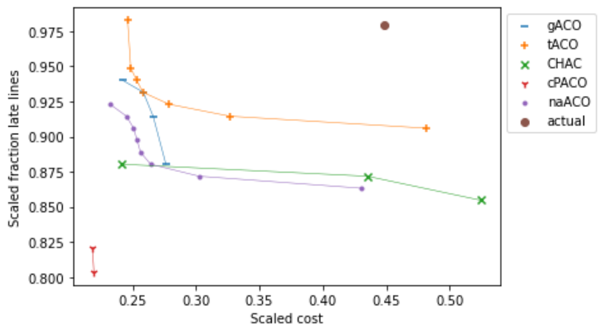
Instance	gACO	tACO	CHAC	cPACO	naACO
1	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
2	0.179 (0.01)	0.168 (0.01)	0.19 (0.01)	**0.251 (0.01)**	0.202 (0.01)
3	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
4	0.122 (0.0)	0.122 (0.0)	0.122 (0.0)	**0.55 (0.0)**	0.122 (0.0)
5	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
6	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
7	0.049 (0.0)	0.048 (0.0)	0.049 (0.0)	**0.055 (0.0)**	0.049 (0.0)
8	0.03 (0.0)	0.028 (0.0)	0.03 (0.0)	**0.032 (0.0)**	**0.032 (0.0)**
9	0.011 (0.0)	0.011 (0.0)	0.011 (0.0)	**0.012 (0.0)**	**0.012 (0.0)**
10	0.067 (0.0)	0.064 (0.0)	0.073 (0.0)	**0.08 (0.0)**	0.076 (0.0)
11	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
12	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
13	0.011 (0.0)	0.011 (0.0)	0.015 (0.0)	**0.019 (0.0)**	0.011 (0.0)
14	0.067 (0.0)	0.067 (0.0)	0.067 (0.0)	**0.068 (0.0)**	0.067 (0.0)
15	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
16	0.086 (0.0)	0.086 (0.0)	0.086 (0.0)	0.086 (0.0)	0.086 (0.0)
17	0.205 (0.01)	0.198 (0.01)	0.216 (0.0)	**0.218 (0.0)**	0.217 (0.0)
18	0.034 (0.0)	0.034 (0.0)	0.034 (0.0)	0.034 (0.0)	0.034 (0.0)
19	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
20	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
21	0.134 (0.0)	0.131 (0.0)	0.147 (0.0)	**0.16 (0.01)**	0.154 (0.0)
22	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
23	0.213 (0.0)	0.207 (0.0)	**0.215 (0.0)**	0.214 (0.0)	**0.215 (0.0)**
24	0.179 (0.0)	0.179 (0.0)	0.179 (0.0)	**0.186 (0.0)**	0.179 (0.0)
25	0.048 (0.0)	0.048 (0.0)	0.048 (0.0)	0.048 (0.0)	0.048 (0.0)
26	0.103 (0.0)	0.099 (0.0)	0.11 (0.0)	**0.125 (0.0)**	0.114 (0.0)
27	0.167 (0.0)	0.166 (0.0)	0.167 (0.0)	0.167 (0.0)	0.167 (0.0)
28	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)
29	0.195 (0.01)	0.182 (0.01)	0.206 (0.01)	**0.222 (0.0)**	0.221 (0.01)

Table 11: Average hyper-volume results of 20 test-runs per test-instance per model. The standard deviation is the number in brackets. The average hyper-volumes of the model that are the best for a given instance are made bold.

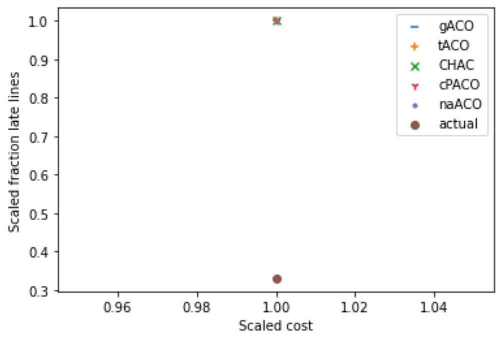
G Figures of all test instance results



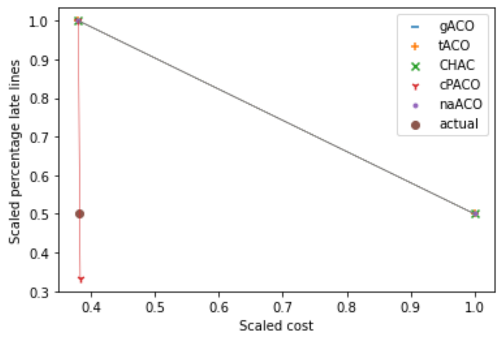
Instance 1



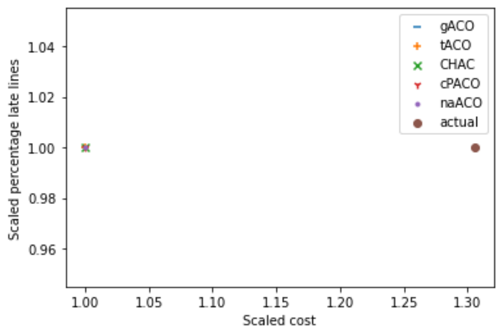
Instance 2



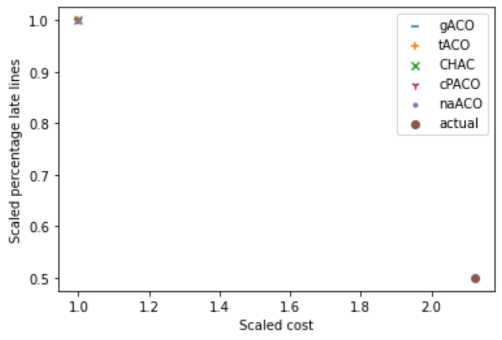
Instance 3



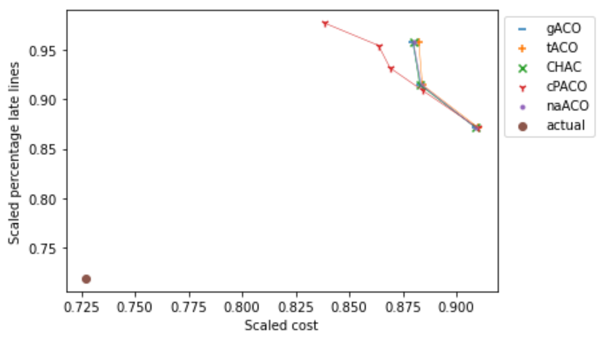
Instance 4



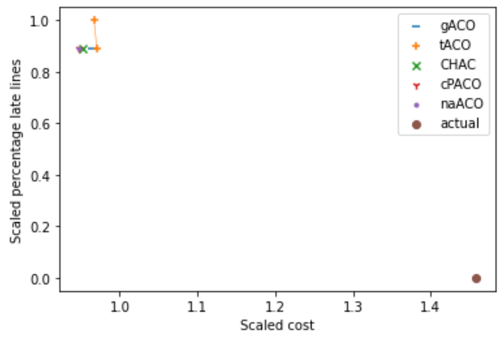
Instance 5



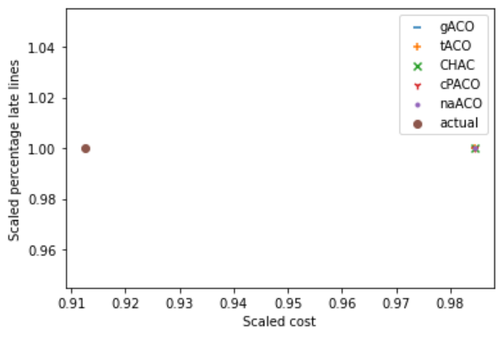
Instance 6



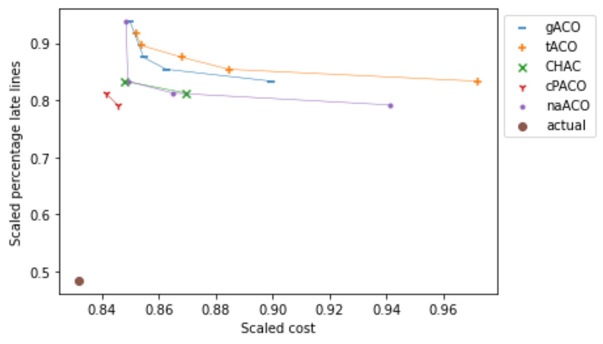
Instance 7



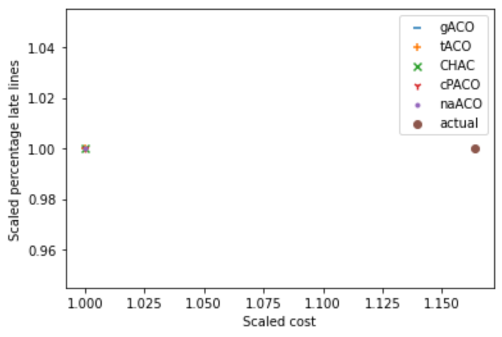
Instance 8



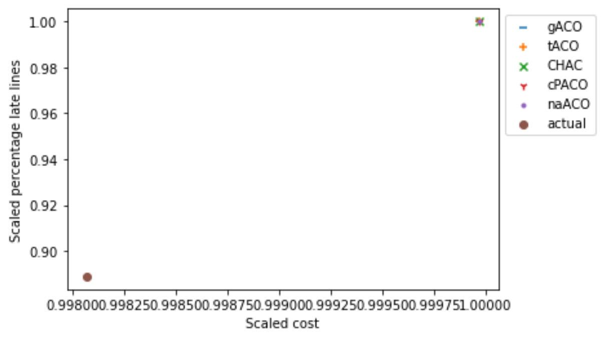
Instance 9



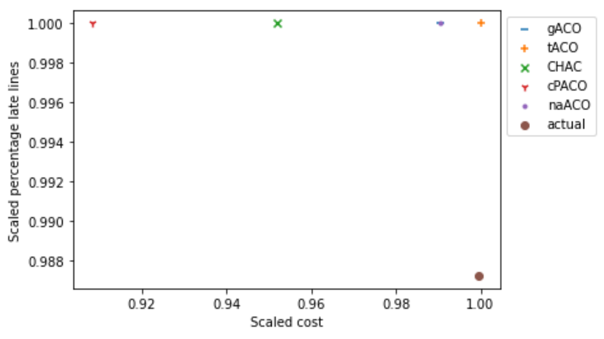
Instance 10



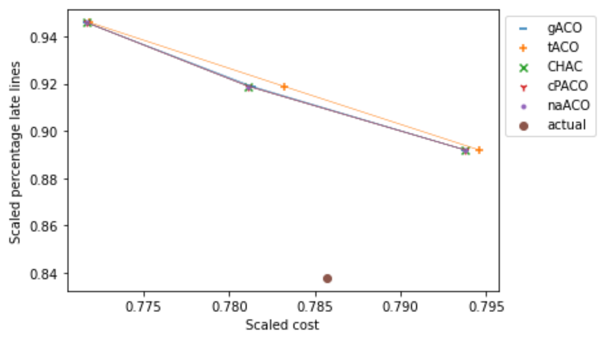
Instance 11



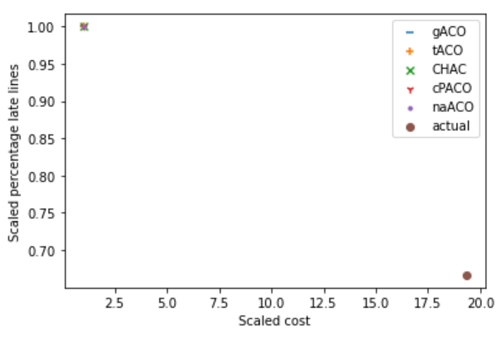
Instance 12



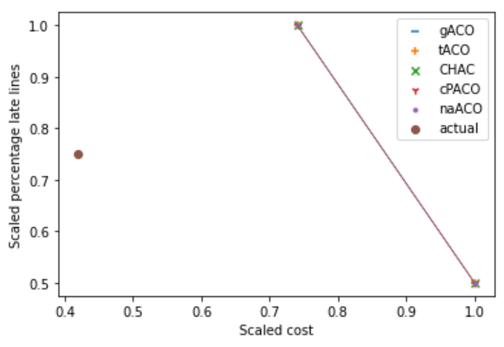
Instance 13



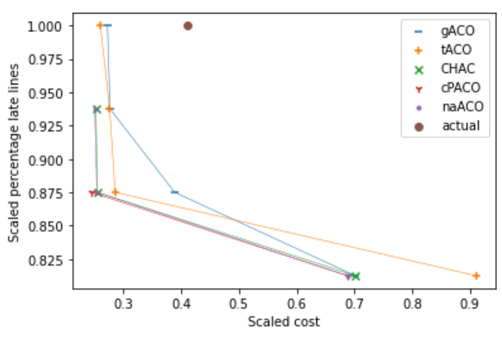
Instance 14



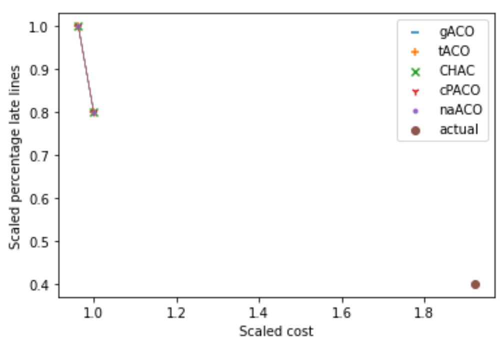
Instance 15



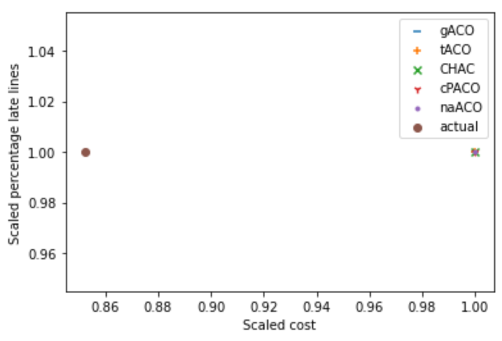
Instance 16



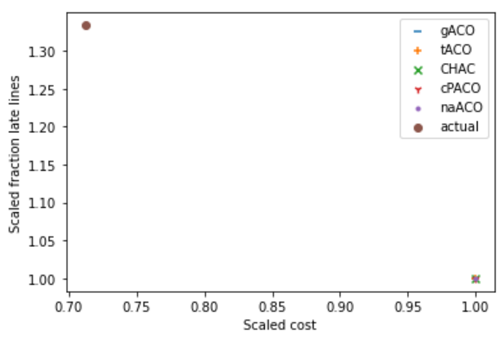
Instance 17



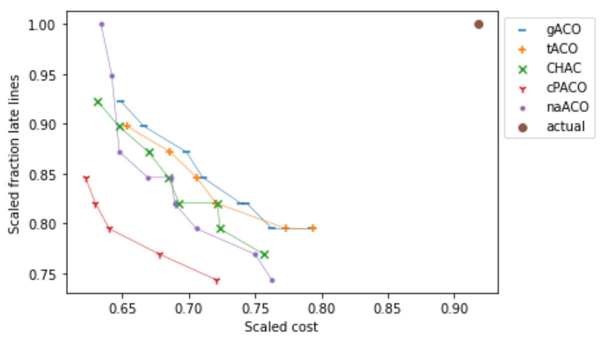
Instance 18



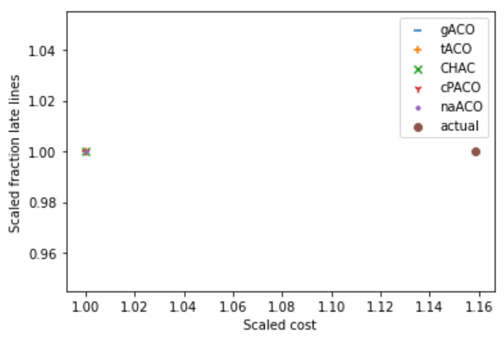
Instance 19



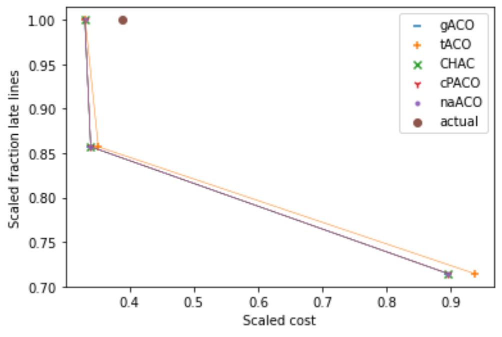
Instance 20



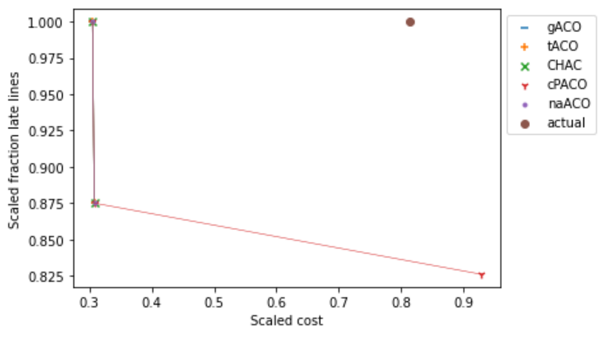
Instance 21



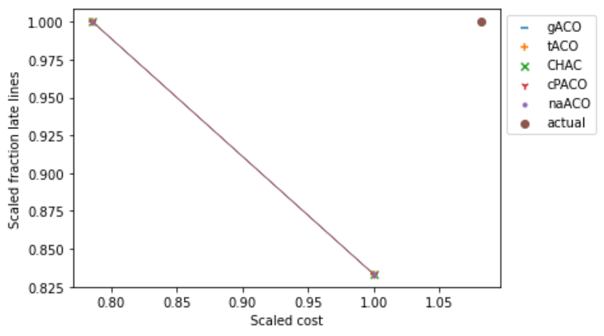
Instance 22



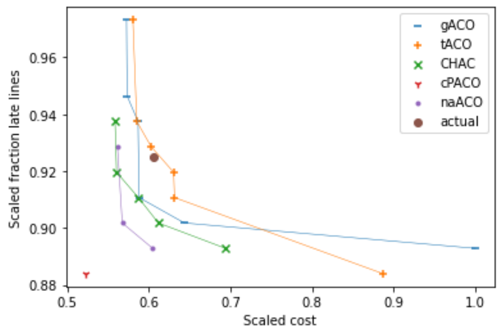
Instance 23



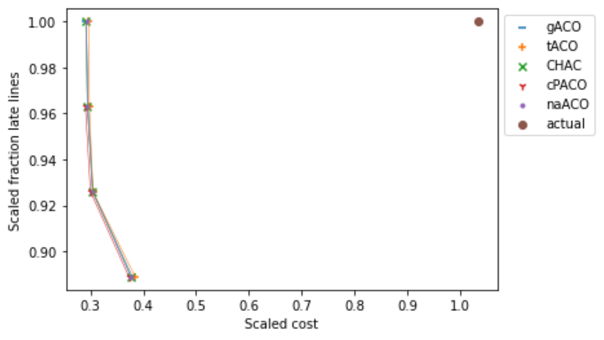
Instance 24



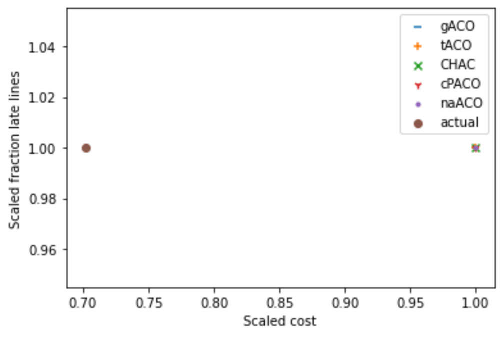
Instance 25



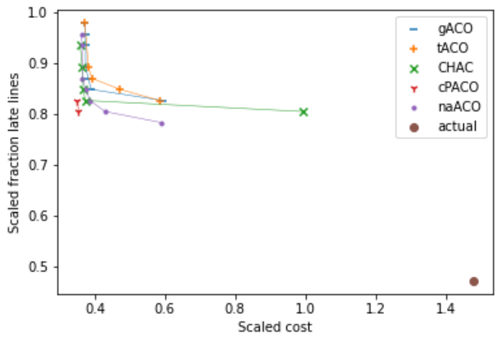
Instance 26



Instance 27



Instance 28



Instance 29

H Cost comparison between methods and actual

	Maximum		Minimum		Average	
Instance	Cost	Percentage difference	Cost	Percentage difference	Cost	Percentage difference
1	1.000	-5.764	1.000	-5.764	1.000	-5.764
2	0.683	52.474	0.236	-47.251	0.310	-30.738
3	1.000	0.000	1.000	0.000	1.000	0.000
4	1.000	162.160	0.380	-0.346	0.690	80.907
5	1.000	-23.412	1.000	-23.412	1.000	-23.412
6	1.000	-52.887	1.000	-52.887	1.000	-52.887
7	0.912	25.404	0.880	21.005	0.891	22.642
8	0.970	-33.438	0.948	-34.978	0.961	-34.11
9	0.999	9.520	0.985	7.897	0.985	7.978
10	1.000	20.173	0.848	1.894	0.870	4.589
11	1.000	-14.080	1.000	-14.08	1.000	-14.080
12	1.000	0.191	1.000	0.191	1.000	0.191
13	0.995	-0.506	0.977	-2.303	0.986	-1.370
14	0.795	1.149	0.772	-1.785	0.781	-0.540
15	1.000	-94.832	1.000	-94.832	1.000	-94.832
16	1.000	138.203	0.74	76.363	0.870	107.283
17	0.902	119.117	0.246	-40.306	0.460	11.659
18	1.000	-47.983	0.963	-49.917	0.981	-48.950
19	1.000	17.337	1.000	17.337	1.000	17.337
20	1.000	40.406	1.000	40.406	1.000	40.406
21	0.968	5.409	0.634	-30.905	0.717	-21.894
22	1.000	-13.694	1.000	-13.694	1.000	-13.694
23	0.947	143.267	0.330	-15.083	0.527	35.374
24	0.308	-62.22	0.305	-62.567	0.306	-62.393
25	1.000	-7.538	0.786	-27.329	0.893	-17.434
26	1.000	65.215	0.555	-8.339	0.630	4.005
27	0.382	-63.089	0.291	-71.885	0.319	-69.158
28	1.000	42.511	1.000	42.511	1.000	42.511
29	0.356	-75.936	0.347	-76.533	0.352	-76.193

Table 12: This table contains the maximum, minimum and average expected cost obtained by gACO for each instance . It also contains the percentage difference between the maximum, minimum and average cost compared to the actual cost.

	Maximum		Minimum		Average	
Instance	Cost	Percentage difference	Cost	Percentage difference	Cost	Percentage difference
1	1.000	-5.764	1.000	-5.764	1.000	-5.764
2	0.856	91.064	0.243	-45.763	0.322	-28.064
3	1.000	0.000	1.000	0.000	1.000	0.000
4	1.000	162.160	0.380	-0.346	0.690	80.907
5	1.000	-23.412	1.000	-23.412	1.000	-23.412
6	1.000	-52.883	1.000	-52.887	1.000	-52.886
7	0.915	25.846	0.880	21.059	0.893	22.858
8	1.000	-31.409	0.954	-34.537	0.971	-33.394
9	1.000	9.587	0.985	7.897	0.990	8.475
10	0.996	19.723	0.846	1.702	0.875	5.210
11	1.000	-14.080	1.000	-14.080	1.000	-14.08
12	1.000	0.194	1.000	0.191	1.000	0.191
13	1.000	0.042	0.979	-2.041	0.991	-0.839
14	0.798	1.507	0.772	-1.785	0.782	-0.482
15	1.000	-94.832	1.000	-94.832	1.000	-94.832
16	1.000	138.203	0.740	76.363	0.87	107.283
17	1.000	142.825	0.246	-40.283	0.473	14.799
18	1.000	-47.983	0.963	-49.917	0.981	-48.95
19	1.000	17.337	1.000	17.337	1.000	17.337
20	1.000	40.406	1.000	40.406	1.000	40.406
21	1.000	8.902	0.625	-31.892	0.722	-21.360
22	1.000	-13.694	1.000	-13.694	1.000	-13.694
23	1.000	156.973	0.330	-15.083	0.541	39.019
24	0.309	-61.997	0.305	-62.567	0.306	-62.376
25	1.000	-7.538	0.786	-27.329	0.893	-17.434
26	0.981	62.139	0.564	-6.830	0.639	5.627
27	0.386	-62.690	0.291	-71.808	0.321	-68.908
28	1.000	42.511	1.000	42.511	1.000	42.511
29	1.000	-32.326	0.365	-75.298	0.465	-68.513

Table 13: This table contains the maximum, minimum and average expected cost obtained by tACO for each instance . It also contains the percentage difference between the maximum, minimum and average cost compared to the actual cost.

Instance	Maximum		Minimum		Average	
	Cost	Percentage difference	Cost	Percentage difference	Cost	Percentage difference
1	1.000	-5.764	1.000	-5.764	1.000	-5.764
2	0.709	58.229	0.232	-48.183	0.304	-32.064
3	1.000	0.000	1.000	0.000	1.000	0.000
4	1.000	162.160	0.380	-0.346	0.690	80.907
5	1.000	-23.412	1.000	-23.412	1.000	-23.412
6	1.000	-52.887	1.000	-52.887	1.000	-52.887
7	0.922	26.814	0.880	21.005	0.891	22.596
8	0.983	-32.56	0.948	-34.978	0.959	-34.221
9	0.999	9.520	0.985	7.897	0.985	7.978
10	0.907	9.022	0.846	1.695	0.861	3.419
11	1.000	-14.08	1.000	-14.080	1.000	-14.08
12	1.000	0.191	1.000	0.191	1.000	0.191
13	0.956	-4.322	0.940	-5.984	0.948	-5.114
14	0.794	1.034	0.772	-1.785	0.782	-0.444
15	1.000	-94.832	1.000	-94.832	1.000	-94.832
16	1.000	138.203	0.740	76.363	0.870	107.283
17	0.720	74.804	0.246	-40.381	0.418	1.416
18	1.000	-47.983	0.963	-49.917	0.981	-48.950
19	1.000	17.337	1.000	17.337	1.000	17.337
20	1.000	40.406	1.000	40.406	1.000	40.406
21	0.856	-6.808	0.623	-32.141	0.694	-24.472
22	1.000	-13.694	1.000	-13.694	1.000	-13.694
23	0.896	130.259	0.330	-15.083	0.522	34.152
24	0.308	-62.220	0.305	-62.567	0.306	-62.393
25	1.000	-7.538	0.786	-27.329	0.893	-17.434
26	0.899	48.469	0.540	-10.789	0.612	1.179
27	0.379	-63.362	0.289	-72.075	0.318	-69.242
28	1.000	42.511	1.000	42.511	1.000	42.511
29	0.992	-32.894	0.358	-75.747	0.411	-72.187

Table 14: This table contains the maximum, minimum and average expected cost obtained by CHAC for each instance . It also contains the percentage difference between the maximum, minimum and average cost compared to the actual cost.

	Maximum		Minimum		Average	
Instance	Cost	Percentage difference	Cost	Percentage difference	Cost	Percentage difference
1	1.000	-5.764	1.000	-5.764	1.000	-5.764
2	1.000	123.209	0.231	-48.392	0.292	-34.916
3	1.000	0.000	1.000	0.000	1.000	0.000
4	1.000	162.160	0.380	-0.346	0.690	80.907
5	1.000	-23.412	1.000	-23.412	1.000	-23.412
6	1.000	-52.887	1.000	-52.887	1.000	-52.887
7	0.910	25.130	0.880	21.005	0.891	22.560
8	0.952	-34.722	0.948	-34.978	0.948	-34.965
9	0.985	7.897	0.985	7.897	0.985	7.897
10	0.941	13.113	0.845	1.500	0.862	3.535
11	1.000	-14.08	1.000	-14.08	1.000	-14.080
12	1.000	0.191	1.000	0.191	1.000	0.191
13	0.993	-0.670	0.979	-2.092	0.987	-1.256
14	0.794	1.034	0.772	-1.785	0.782	-0.443
15	1.000	-94.832	1.000	-94.832	1.000	-94.832
16	1.000	138.203	0.740	76.363	0.870	107.283
17	0.693	68.224	0.245	-40.437	0.417	1.210
18	1.000	-47.983	0.963	-49.917	0.981	-48.950
19	1.000	17.337	1.000	17.337	1.000	17.337
20	1.000	40.406	1.000	40.406	1.000	40.406
21	0.849	-7.595	0.623	-32.186	0.695	-24.288
22	1.000	-13.694	1.000	-13.694	1.000	-13.694
23	0.896	130.259	0.330	-15.083	0.522	34.152
24	0.308	-62.220	0.305	-62.567	0.306	-62.393
25	1.000	-7.538	0.786	-27.329	0.893	-17.434
26	0.972	60.652	0.546	-9.760	0.604	-0.254
27	0.378	-63.449	0.289	-72.017	0.317	-69.343
28	1.000	42.511	1.000	42.511	1.000	42.511
29	0.946	-35.966	0.356	-75.917	0.407	-72.472

Table 15: This table contains the maximum, minimum and average expected cost obtained by naACO for each instance . It also contains the percentage difference between the maximum, minimum and average cost compared to the actual cost.