Erasmus University Rotterdam

Economics & Informatics

Erasmus School of Economics

Master Thesis

Evolutionary Multi-Objective Optimization and Preference Modeling in Green Logistics

Author:
Corné van der Plas (306055)
vdplas@gmail.com

Supervisor
Dr. Tommi Tervonen
tervonen@ese.eur.nl

Second Supervisor
Prof.Dr.Ir. Rommert Dekker
rdekker@ese.eur.nl

July 9, 2012

Abstract

This master thesis considers supply chain design in green logistics. The research consists of two parts. First, we formulate the environmentally conscious design as a a multi-objective optimization problem and construct the Pareto front using scalarizing methods (weighted sum and ε-constraint method) and genetic algorithms (NSGA-II/SPEA2). The second part involves constructing a preference model to aid the decision maker (DM) in choosing the preferred alternative using the UTA$^{\text{GMS}}$ method.

This research includes a case study for a supply chain in the South Eastern Europe region; it extends the work of I. Mallidis, R. Dekker, and D. Vlachos. The impact of greening on supply chain design and cost: a case for a developing region. *Journal of Transport Geography*, 22:118–128, 2012. We apply a genetic algorithm to optimize simultaneously cost, CO_2 emission and Particulate Matters (PM – also known as fine dust), and to present a set of alternatives to the DM (the Pareto front). In this case study there are two different scenarios: both have the distribution centers outsourced, one also outsources the transportation while the other scenario leases the transportation. First we compare the different method to see which method give the best representation of the Pareto front. Then the UTA$^{\text{GMS}}$ method will be used to aid the DM in choosing his/her most preferred solution. For the UTA$^{\text{GMS}}$ method, the DM is asked to provide his/her preference information by means of pairwise comparisons. Some computational tests are used to determine how applicable the UTA$^{\text{GMS}}$ method is to this particular case.

Contents

Chapter 1

Introduction

Green logistics has become more popular over the last years. Due to the liberalization of international trade, extensive and complex supply chain networks with various demand points around the world are developed. For these supply chains, cost minimization is often the most important optimization objective. However, in green logistics also other objectives are considered, namely ones that quantify the environmental impact. The emission of greenhouse gases has increased over the last years due to the growth of international transport, and should be taken into account in supply chain design for their negative influence on the environment. Environmentally friendlier transportation has been promoted by governmental initiatives such as the Kyoto protocol and European Union action plans including (i) Freight Transport Logistics Action Plan (2007) [19], (ii) Greening Transport (2008) [20], (iii) Strategy for the internalization of external costs (2008) [21], and (iv) A sustainable future for transport: Towards an integrated, technology led and user friendly system (2009) [22]. Therefore, optimization of supply chains nowadays also has to consider, besides the cost, the environmental aspects such as the emission of carbon dioxide (CO_2) and particulate matters (PM, also known as fine dust).

In this research we introduce a case for optimizing simultaneously costs, CO_2 and PM emissions while designing a supply chain of white goods for South Eastern Europe. The original model, which only solves the three objectives individually, was proposed by Mallidis et al. [46]. We will optimize the multi-objective mixed integer linear programming (MOMILP) problem by applying two scalarization methods and two evolutionary algorithms. After obtaining a set of possible solutions, we apply a preference model to order (rank) the solutions. Because of the increasing importance of the environmental impact of the supply chain, it would be interesting to see whether these changes in preferences will lead to intermediate solutions in which not only the cost is optimized and how much the cost may increase in order to reduce the emission of CO_2 or PM.

In Multi-Objective Optimization (MOO) there does not exists just one optimal solution like in single-objective optimization problems. MOO deals with contradictory objectives and the 'best' solution should make a compromise between them. Without any further information you cannot say that one solution is better than one other solution, unless one dominates the other. One solution dominates the other if it has at least one better objective value while the other objectives are at least the same (or better) related to a dominated solution. Thus you are interested in all non-dominated solutions, called the Pareto front, to have all alternatives

available to determine what solution is the best.

There are three approaches to involve the preferences of the decision maker (DM), namely *a priori*, interactive or *a posteriori*. In the first case the preference information will be used directly within the solving process. In the second case, during an interactive approach based on pairwise comparisons, the DM will be guided to his/her most preferred solution. In the third case, the whole set of solutions will be generated and then a preference model can be constructed to aid the DM. In this research we only take the latter in consideration. We will apply the UTA$^{\text{GMS}}$ method [29] that builds a set of additive value functions to obtain a complete preorder of the set of alternatives based on preference statements.

There exists many approaches to obtain the Pareto set of solutions for a MOO model. Three commonly used methods that scalarize the problem into a single objective are objective weighting, distance functions and a method of min-max formulation [61]. However, these three approaches convert the MOO problem to a single-objective optimization problem and thus result in a single solution. One can use scalarizing approaches to generate multiple solutions. Although these methods generate multiple solutions, they do not find the whole Pareto front, e.g., the weighted sum method cannot find solutions that lie in non-convex regions of the Pareto front. Evolutionary algorithms have been shown to provide good approximations of the Pareto front [see 15, 24, 61].

1.1 Problem description

The base model is described by Mallidis et al. [46]. This paper describes the design of the supply chain network for the South Eastern Europe region and the impact different optimization criteria have on cost and design of the network. In this model they only consider single-objective optimization, with four different scenarios in different models. They analyze the impact of leasing or outsourcing transportation, and also leasing or outsourcing distribution centers. In this research we extend the work of Maladis et al. by applying multi-objective optimization with scalarization. We will discuss these results in section 4.4.1.

We will only include the solutions for outsourcing a distribution center as it seems that outsourcing a distribution center is always cheaper and has less emissions than leasing it (see results in [46]) when outsourcing is feasible (e.g., it may not be feasible when long term contracts are required). It is obvious that you only include non-dominated solutions in the decision making process. In case of leasing a distribution center (DC), [46] is not clear about what the minimum throughput should be to be profitable and thus to open the DC. Therefore, we will not consider the possibility of leasing a distribution center. However, given a set of possible solutions, which alternative is the best? Different preferences leads to different choices, and we want to build a model supporting the choice of an alternatives over another one.

Probably, the weighted sum and ε-constraint method are not able to provide a good approximation of the Pareto front. Therefore, we need to apply a method that does generate (a good approximation of) the Pareto front before we apply our preference model. The solution set has to be complete in order to know whether the most preferred solution is really the best solution for this DM. We will also compare the Pareto front obtained with the more complex method to

that of the weighted sum and ε-constraint methods. Then we can see whether the efforts of the complex method leads to a better estimate of the Pareto front or whether a simpler method like the weighted sum method is also appropriate in terms of generating the Pareto front.

1.2 Research question

In this research we apply a genetic algorithm to generate (or approximate) the Pareto front for a multi-objective optimization model. Then we will apply a preference model according to the UTA$^{\text{GRIP}}$ method in order to help the decision maker (DM) to find the best solution. We can formulate the following research question:

Can we aid the DM to design a supply chain in green logistics, by representing alternatives obtained with genetic algorithm and thereafter apply the UTA^{GMS} method?

In order to answer this research question, we have the following sub-questions:

1. How different is the Pareto front obtained with a genetic algorithm (NSGA-II/SPEA2) related to one from the weighted sum method?
2. How applicable is the UTA$^{\text{GMS}}$ method to this specific multi-objective optimization model?
3. How many preference statements do we need in order to obtain a single (best) solution?

Although there exists multiple algorithms to generate the Pareto front, this research is limited to applying the NSGA-II and SPEA2 algorithm, together with the UTA$^{\text{GMS}}$ method. The main focus will be to apply the UTA$^{\text{GMS}}$ method to the MOO context and perform computational tests to determine how many preference statements are needed to build a preference model, and how much computational time is required in order to find a single 'best' solution for the DM.

We will build a model for optimizing the supply chain and advising the DM a solution based on his/her preferences. The first part, thus the genetic algorithm, will generate a set of possible decision alternatives. Then, the UTA$^{\text{GMS}}$ method will be used to aid the DM to find the most preferred solution among these alternatives.

1.3 Methodology

The generation, or at least a good approximation, of the Pareto front can be obtained by applying a genetic algorithm. In this research we will apply the Non-dominated Sorting Genetic Algorithm (NSGA-II) and the Strength Pareto Evolutionary algorithm 2 (SPEA2) to approximate the Pareto front. These two algorithms do not involve any preference information, so there is a clear separation between obtaining the set of non-dominated solutions and applying a preference model to rank them. Both algorithms are able to generate a good representation of the Pareto front for a multi-objective optimization problems, and we can test which one suits this particular problem better. Also, these approaches will be compared to the scalarization methods to see

whether genetic algorithms do a better job than the 'simpler' scalarization methods or whether scalarization methods are also sufficient.

The UTA^{GMS} method will be used as a procedure to deal with the DM's preferences. This method is proposed by Greco et al. [29] and is used for ranking a finite set of actions evaluated on multiple criteria. The method builds a set of additive value functions based on the DM's preferences and besides the preference relation. The UTA^{GMS} method itself is an extension of the UTA method with ordinal regression (see [37]).

The NSGA-II and SPEA2 will be implemented using the ParadisEO framework [12, 44]. That is a object-oriented framework in C++ for the reusable design of metaheuristics. The framework provides the MOO algorithms NSGA, NSGA-II, IBEA and SPEA2. To test these results, we compare solutions of genetic algorithms with those of the scalarization methods. Then the preference model will be implemented using R [52], a free software environment for statistical computing, together with the 'ror' library[1].

To determine how valuable our preference model can be in practice, we perform computational tests to investigate how many and what kind of preference statements are needed in order to build a preference model. The UTA^{GMS} method needs preference information in form pairwise comparisons. Because there is no real DM involved, we will execute tests with random preference statements to determine the applicability of our preference model in the particular case study.

1.4 Structure

The thesis is organized as follows. In Chapter 2 we will describe the applied approaches for estimating the Pareto front for a multi-objective optimization model: scalarization methods and genetic algorithms. We will discuss preference models and the related theory in Chapter 3. In Chapter 4 we describe the case study and their results. In Chapter 5 we discuss the methods that are used and then in Chapter 6 we present the conclusions of this research.

[1]Robust Ordinal Regression `http://cran.r-project.org/web/packages/ror/`

Chapter 2

Multi-Objective Optimization

Multi-objective optimization (MOO) is the process of simultaneously optimizing multiple conflicting goals or objectives. The difference with single-objective optimization is that MOO results in several/many equally efficient solutions, known as Pareto optimal solutions, instead of one single solution. A MOO problem consists of a set of objective functions that can be either maximized or minimized. A set of constraints limits the set of possible outcomes, known as the solution space. Without loss of generality, a MOO problem can be defined as follows (we assume all objective functions to be minimized):

$$\begin{aligned} \text{Minimize} \quad & F(x) = \big(F_1(x), F_2(x), \ldots, F_n(x)\big) \\ \text{subject to} \quad & g_j(x) \geq 0,\ j = 1, \ldots, J. \\ & h_k(x) = 0,\ k = 1, \ldots, K. \end{aligned} \tag{2.1}$$

In [14] two goals of MOO are described: (i) convergence to the Pareto optimal solutions and (ii) maintenance of a set of maximally-spread Pareto optimal solutions. Both goals are independent to each other and the optimization algorithm must have properties to achieve both goals. MOO results in multiple solutions while a user often only needs one solution. In [14] there are two steps defined in order to find a single (best) solution:

1. Find multiple trade-off optimal solutions with a wide range of values for objectives;
2. Choose one of the obtained solutions using higher-level information.

The ideal way, according to Deb [14], is to find first a well-distributed set of trade-off solutions (step 1) and then apply the preferences (step 2). This way is less subjective than using first the preference information. In the latter, one can estimate a relative importance vector using higher-level information, i.e. the DM's preference information, which results in a single solution. Therefore, in this research we will first focus on the Pareto-set and thereafter apply a preference model.

In most MOO algorithms is the concept of dominance used to compare two solutions. A solution dominates another solution if the following two conditions are met:

1. The solution is no worse than the other solution in all objectives;
2. The solution is strictly better than the other solution in at least one of the objectives.

Without any preference information and all objectives being equally important, we cannot say that one solution is better than the other solution unless it dominates the other. Among a set of solutions, the set of non-dominated solutions is called the Pareto optimal set or Pareto front. The Pareto front represents the solutions that you cannot improve on one of the objectives without worsening another objective.

There are different approaches to optimize MOO problems; below we describe some of them. First we describe scalarization methods, then genetic algorithms, and finally other approaches.

2.1 Scalarization methods

Scalarization methods are methods that scalarize a multi-objective problem to a single-objective problem. Ehrgott [18] defines the principle of scalarization to solve a MOO problem as 'converting the multi-objective program to a single-objective program that usually depends on some parameters not included in the objective function and then solve the scalarized problem repeatedly with different parameter values'. Below we describe the Weighted Sum and ε-Constraint Methods, and other scalarizing approaches.

2.1.1 Weighted Sum Method

The weighted sum method assigns a non-negative weight to each objective and normally the weights sum up to one. The mathematical definition is shown in Equation 2.2. The different objectives do not have to be scaled because the weights merely serve to find solutions on the Pareto front. By changing the set of weights, a different point on the Pareto front can be obtained. However, there are three difficulties with the weighted sum method [13]: (i) there is no satisfactory (a priori) selection method to determine the weights that guarantee the final solution to be acceptable, (ii) it cannot find solutions on non-convex regions of the Pareto front, and (iii) varying the weights may not result in an evenly distributed and accurate/complete representation of the Pareto front.

$$\begin{array}{lll} \text{Minimize} & F(x) = \sum_{m=1}^{M} w_m f_m(x) & \\ \text{Subject to} & g_j(x) \geq 0 & j = 1, 2, ..., J \\ & h_k(x) = 0 & k = 1, 2, ..., K \end{array} \tag{2.2}$$

There exist extensions of the weighted sum method for dealing with these three issues. Kim and de Weck [41] presented the adaptive weighted sum method, which is an improvement to their earlier presented bi-objective adaptive weighted sum method [40]. They demonstrate that their adaptive weighted sum method finds solutions for non-convex regions and a well-distributed Pareto front. After normalization and applying the usual weighted sum method, this method involves some additional steps. First, nearly overlapping solutions are removed. Then Pareto front patches have to be identified which will be further refined, and then using additional equality constraints and sub-optimization for these patches, new solutions are obtained. Finally

a Pareto-filter is applied to remove all dominated solutions.

2.1.2 ε-Constraint Method

To overcome the difficulty that the weighted sum method does not find solutions in a non-convex region, the ε-Constraint Method is used. This method optimizes one objective, while the other objectives are used as constraints. Consequently, compared with the weighted sum method (Equation 2.2) there is only one objective function ($f_\mu(x)$) and additional constraints ($f_m(x) \leq \varepsilon_m$) that require the other objectives do not exceed the user-defined values. The new equation is shown in Equation 2.3.

$$\begin{array}{lll} \text{Minimize} & f_\mu(x), & \\ \text{Subject to} & f_m(x) \leq \varepsilon_m & m = 1, 2, ..., M \text{ and } m \neq \mu \\ & g_j(x) \geq 0 & j = 1, 2, ..., J \\ & h_k(x) = 0 & k = 1, 2, ..., K \end{array} \tag{2.3}$$

One difficulty about this method is how to choose the ε_m values; you do not know beforehand what the best values will be. If you increase the ε-values with too small steps it leads to a lot of redundant runs, and if the steps between different runs are too large it misses Pareto optimal solutions. Thus it requires more user input to find solutions in non-convex regions, but the ε-Constraint method has also several advantages over the Weighted Sum Method:

1. The Weighted Sum Method only finds extreme points for linear models, thus a lot of runs are redundant because they result in the same solution. The ε-Constraint Method is able to produce non-extreme efficient solutions.

2. The Weighted Sum Method cannot find solutions for non-convex regions, while the ε-Constraint Method does not suffer from this pitfall.

3. The ε-Constraint Method can control the number of efficient solutions, while this is not so easy for the Weighted Sum Method.

Research that has been done to improve the ε-constraint method, e.g. [43] and [47]. The former presents an adaptive scheme that finds appropriate constraint values during the run. The latter [47] proposes a novel version of the ε-constraint method: the augmented ε-constraint method (AUGMECON). This is an effective implementation of the ε-constraint method using lexicographic optimization for the payoff table, and can be used in an interactive context.

2.1.3 Other

A third scalarization method minimizes the distance to an ideal point (i.e., the solution in which all objectives are minimized simultaneously). However, the ideal point is not a feasible solution when there are conflicting objectives. The compromise programming method applies this principle, and one example shown in [18] is the following minimization problem (with a weighting vector $\lambda > 0$, an integer $1 \leq q < \infty$, ideal point y^I and $x \in X$ representing the

constraints):

$$\min \left\{ \left(\sum_{k=1}^{p} \lambda_k \left(f_k(x) - y_k^I \right)^q \right)^{\frac{1}{q}} : x \in X \right\} \tag{2.4}$$

In [48] are a number of scalarization functions described; these are STEM, STOM, achievement scalarizing functions, GUESS, NIMBUS, a lexicographic formula and four variants of integer-valued multi-objective optimization. These functions are either reference-point based or used for classification. In the latter the decision maker is asked to classify a set of current solutions to be acceptable or whether some objectives need improvement. Thus the functions based on classification already take preferences into account.

2.2 Genetic Algorithm for MOO

A genetic algorithm is a search heuristic and belongs to a larger class called evolutionary algorithms. Genetic algorithms normally deal with population sizes larger than one. Each individual of the population represents a single solution, and therefore genetic algorithms seem well suited for generating the optimal solution set in multi-objective optimization problems.

Genetic algorithms are based on Darwin's theory about evolution. A genetic algorithm start with an initial set of solutions, called the population, represented by chromosomes. In each generation, new solutions are generated using genetic operators such as recombination, crossover and mutation. This way you try to find better solutions by improving earlier ones. For each individual in the population (in each generation), a fitness value $F(x)$ is calculated representing the goodness of the solution. The algorithm stops when an end condition is satisfied.

The basic principles of a standard genetic algorithm consist of coding, a fitness function and reproduction. The coding means the representation of the potential solution in terms of a string of values (called chromosomes). Often the binary alphabet is used, but it depends on what is best suited for the particular problem. The fitness function returns a numerical fitness (or 'utility') value for a particular chromosome. This indicates how good the potential solution is. In the reproduction phase, individuals from the population are selected and recombined to generate the offspring for the next generation. The recombination is typically done with mechanisms like crossover and mutation. Crossover takes two individuals and uses random point(s) to cut the chromosome in two segments, a 'head' and 'tail' segment [7]. The tail segments are swapped over to produce two new chromosomes. Figure 1 shows an example of a single-point crossover. Usually, crossover is not applied to all pairs of chromosomes, but has a likelihood of being applied typically between 0.6 and 1.0. Often the fittest chromosomes are selected for crossover, but one can also use different selection mechanisms, e.g., tournament selection. With tournament selection, the individual with the best fitness has the highest chance of being chosen. For example, the best individual can have p chance to be chosen, the second best $p \times (1-p)$, the third $p \times ((1-p)^2)$, and so on. Mutation is applied, after crossover, to each child instead of pairs. It randomly alters a gene and is typically applied in 0.01 of the cases to escape from local optima. Figure 2 shows a single-point mutation.

Fonseca and Fleming [26] describe four different approaches for evolutionary algorithms

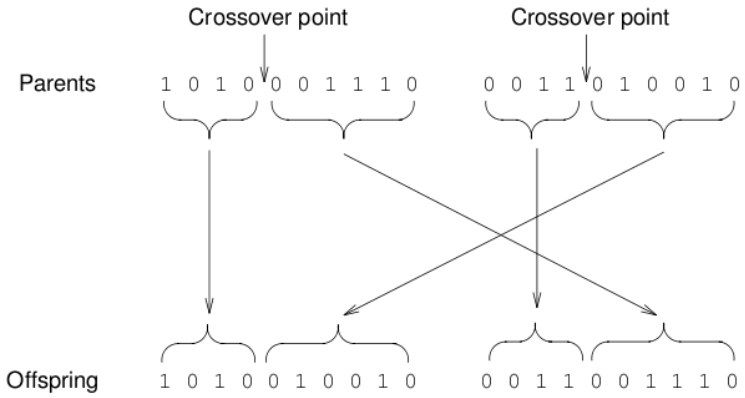


Figure 1: Crossover [7]

Figure 2: Mutation [7].

in multi-objective optimization: (i) plain aggregation approaches, (ii) population-based non-Pareto approaches, (iii) Pareto-based approaches, and (iv) niche induction techniques. In the first approach the multiple objectives are aggregated into a scalar function. These methods are similar to the weighted sum approach described in Section 2.1.1. The second approach was the first approach that optimizes multiple objectives simultaneously, like the Vector Evaluated Genetic Algorithm (VEGA) method from Schaffer [56], but does not make direct use of Pareto optimal solutions, i.e., all Pareto optimal solutions have the same fitness. At most, they monitor the population for non-dominated solutions. The third approach ranks the population based on their non-domination. The solutions that have the same rank get the same fitness value, thus all Pareto optimal solutions have the same fitness. This approach does not guarantee that the Pareto-set is uniformly sampled, unlike the last approach. Niche induction techniques have the additional use of fitness sharing and can also add a mating restriction. Both are necessary to produce a distance measure needed for uniform sampling.

2.2.1 NSGA-II

The Non-dominated Sorting Genetic Algorithm ([16]) has the following three properties:

1. It uses an elitist principle;
2. It uses an explicit diversity preserving mechanism;
3. It emphasizes the non-dominated solutions.

For the fast non-dominated sorting approach two entities need to be computed: (i) domination count n_p, the number of solutions which dominate the solution p; and (ii) S_p, a set of solutions that the solution p dominates. The solutions with $n_p = 0$ represent the first non-dominated front. Then, for each solution with $n_p = 0$ (thus from the first non-dominated front), we visit each member (q) of its set S_p and reduce its domination count by one (thus we remove solution p from n_q). For any member for which domination count becomes zero ($n_q = 0$), we put in a separate list Q. Then Q represents the second domination front. These procedures are repeated for each member of Q to identify the third front, and we continue until all fronts are identified.

To obtain a density estimation of solutions surrounding a particular solution, we compute the average distance of two points on either side of the point along each of the objectives. This quantity $i_{distance}$ serves as an estimate of the perimeter of the cuboid formed by using the nearest neighbors as the vertices (this is the crowding distance). In Figure 3 is the crowding-distance of the i-th solution in its front (marked with filled circles) the average side length of the cuboid (shown with a dashed box). The following algorithm is used to calculate the crowding-distance for each point in set I:

1. Call the number of solutions in I as $l = |I|$. For each i in the set, first assign $I[i]_{distance} = 0$;
2. For each objective m, sort the set in ascending order;
3. For each objective m, assign a large distance to the boundary solutions, or $I[1]_{distance} = I[l]_{distance} = \infty$, and for all other solutions $i = 2$ to $(l - 1)$, assign

$$I[i]_{distance} = I[i]_{distance} + \frac{I[i+1]_m - I[i-1]_m}{f_m^{max} - f_m^{min}}$$

Thus the crowding-distance computation requires first to sort the population in ascending order for each objective. Then for each objective function are the boundaries set to infinity, and for all other (intermediate) solutions the distance is the absolute normalized difference in the function values of two adjacent solutions. The last is repeated for all other objectives. Then the total crowding-distance is the sum of individual distance values corresponding to each objective, with each objective being normalized.

The crowded-comparison operator ($\succ_n$) ensures a uniform spread-out of the Pareto front during the various stages of the algorithm. Assume that every individual i has the following two attributes: non-domination rank (i_{rank}), and crowding distance ($i_{distance}$). Then the partial preorder $\succ_n$ is defined as

$$i \succ_n j \text{ IF } (i_{rank} < j_{rank}) \text{ OR}$$
$$\Big((i_{rank} = j_{rank}) \text{ AND } (i_{distance} > j_{distance})\Big)$$

Thus in order for a solution to be preferred to another one in terms of necessarily better, it needs a better rank (a better non-domination front) or a better crowding distance in case of the same rank.

The main loop starts with the initialization of a random parent population P_0 sorted based on the non-domination. First the offspring Q_0 of size N will be created using the usual binary

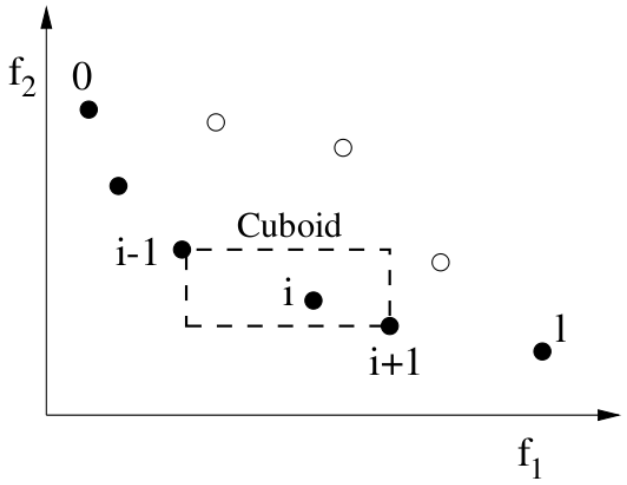


Figure 3: Crowding-distance calculation [16]

.

tournament selection, recombination and mutation operators. Algorithm 1 describes the procedure for the t-th generation. First we combine the parents and the offspring ($R_t = P_t \cup Q_t$), which has the size $2N$. Then we sort the population R_t according to their non-domination. Elitism is ensured because the current as well as the previous members are included in R_t. The new population (P_{t+1}) will be filled with the best fronts (first $\mathcal{F}_1$, then $\mathcal{F}_2$, etc.), until the size of the next front ($\mathcal{F}_l$) is bigger than the number of open spots in P_{t+1}. To have exactly N members in the new population and the diversity preservation (i.e., that a good spread of solutions is maintained in the obtained solution set), the front $\mathcal{F}_l$ will be ordered based on the crowding-distance and the first $N - |P_{t+1}|$ (i.e., the number of open spots) solutions will be added to end up with exactly N solutions in P_{t+1}. Then we start again to make an offspring (Q_{t+1}) of P_{t+1} and we repeat this algorithm until the stopping criterion is met.

Algorithm 1 NSGA-II (main-loop for the t-th generation)

$R_t = P_t \cup Q_t$
$\mathcal{F} =$ `fast-non-dominated-sort`$(\mathtt{R_t})$
$P_{t+1} = \emptyset$ and $i = 1$
while $|P_{t+1}| + |\mathcal{F}_i| \leq N$ **do**
  `crowding-distance-assignment`$(\mathcal{F}_\mathtt{i})$
  $P_{t+1} = P_{t+1} \cup \mathcal{F}_i$
  $i = i + 1$
end while
Sort$(\mathcal{F}_i, \succ_n)$
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i[1 : (N - |P_{t+1}|)]$
$Q_{t+1} =$ `make-new-pop`$(\mathtt{P_{t+1}})$
$t = t + 1$

2.2.2 SPEA2

The Strength Pareto Evolutionary Algorithm 2 (SPEA2) is an improvement over the earlier presented algorithm SPEA, by Zitzler and Thiele [65]. SPEA2, proposed by Zitzler et al. [66], takes new features into account. The main differences with its predecessor are:

- An improved fitness assignment for each individual, which takes into account how many solutions it dominates and how many solutions are dominated by;
- A nearest neighbor density estimation technique to have better guidance of the search process;
- New archive truncation methods that guarantee the preservation of boundary solutions.

We describe the basis of SPEA first and then we show the improvements made by SPEA2. SPEA starts with an initial population and an empty set called the archive. Then for each iteration the following steps are performed. First are all non-dominated solutions from the population copied to the archive. Then all duplicates or dominated solutions from the archive are removed. If the resulting archive is greater than a predefined maximum size, further archive members are removed by a clustering technique. Then for each individual in both the population and archive sets the fitness is calculated:

- Each individual in the archive set is assigned a strength value $S(i) \in [0, 1)$, which represents the fitness $F(i)$ at the same time. Thus for each non-dominated solution (or archive member), the fitness is equal to its strength. $S(i)$ is the number of dominated solutions or equal solutions (j) by i divided by the population size (N) plus one;
- The fitness of $F(j)$ is calculated by summing up $S(i)$ for all archive members i that dominates or are equal to j and adding one at the end.

This means that for all non-dominated solutions the fitness is between 0 and 1 ($0 < F(j) < 1$) and all dominated solutions have a fitness value higher than one ($F(j) > 1$). Then the next step is the mating selection phase, where individuals from the union of the population and the archive are selected by means of binary tournament selection (i.e., the better the individual, the higher chance to be chosen). Finally, recombination and mutation result in new offspring population.

In contrast to SPEA, SPEA2 uses a different fitness assignment: a fixed archive size, the clustering technique is replaced by an alternative truncation method, and only archive members participate in the mating selection phase. To avoid situation that population members dominated by the same members of the archive have the same fitness value, both the number of dominating and dominated solutions are taken into account. The raw fitness is determined by the number of its dominators in both the archive and population, where in SPEA only the archive members are considered. The final fitness value is composed of this raw fitness value plus additional density information. If you take just the raw fitness as the final fitness value, it may fail when most individuals do not dominate each other. Therefore, an adaptation of the k-nearest neighbor is used for additional density information to discriminate solutions that have the same raw fitness value.

A fixed size used for the archive in the SPEA2 environmental selection. Like in SPEA, first are all solutions that are non-dominated copied, i.e. all solutions that have a fitness value smaller than one. If the size of the archive is exactly the correct one ($|\bar{P}_{t+1}| = \bar{N}$), the environmental selection is completed. Else there can be two situations: the archive is either too small or too big. In the first case, the best dominated solutions in the previous archive and population are copied to the new archive. In the latter case, an archive truncation procedure is invoked that iteratively removes solutions until the size is equal to the predefined size. The ones that have the minimum distance, defined with the k-th nearest neighbor procedure, to other solutions will be removed. Algorithm 2 shows an overview of the SPEA2 procedure.

Algorithm 2 SPEA2 Algorithm [66]

Input:	N	*population size*
	$\bar{N}$	*archive size*
	T	*maximum number of generations*
Output:	$\boldsymbol{B}$	*non-dominated set*

Step 1: **Initialization:** Generate an initial population $\boldsymbol{P}_0$ and create the empty archive (external set) $\bar{\boldsymbol{P}}_{\boldsymbol{0}} = \emptyset$. Set $t = 0$.

Step 2: **Fitness assignment:** Calculate fitness values of individuals in P_t and $\bar{P}_t$

Step 3: **Environmental selection:** Copy all non-dominated individuals in P_t and $\bar{P}_t$ to $\bar{P}_{t+1}$. If size of $\bar{P}_{t+1}$ exceeds $\bar{N}$ then reduce $\bar{P}_{t+1}$ by means of the truncation operator, otherwise if size of $\bar{P}_{t+1}$ is less than $\bar{N}$ then fill $\bar{P}_{t+1}$ with dominated individuals in P_t and $\bar{P}_t$

Step 4: **Termination:** If $t \geq T$ or another stopping criterion is satisfied then set A to the set of decision vectors represented by the non-dominated individuals in $\bar{P}_{t+1}$. Stop.

Step 5: **Mating selection:** Perform binary tournament selection with replacement on $\bar{P}_{t+1}$ in order to fill the mating pool.

Step 6: **Variation:** Apply recombination and mutation operators to the mating pool and set P_{t+1} to the resulting population. Increment generation counter ($t = t + 1$) and go to Step 2.

2.2.3 Other genetic algorithms

The NSGA [61] was one of the first methods that solve multiple objectives simultaneously. In this algorithm, the population is first sorted (ranked) based on an individual's non-domination, and a sharing method is used to keep the diversity within the population after reproduction. The improved version (NSGA-II), addresses the three main criticisms of NSGA: (i) high computational complexity of non-dominated sorting, (ii) lack of elitism, and (iii) need for specifying the sharing parameter. The original algorithm only varies in the way the selection operator works.

Zitzler and Thiele [65] presented the SPEA algorithm. This algorithm combined features from four older algorithms: (i) the Vector Evaluated Genetic Algorithm (VEGA) from Schaffer [56]; (ii) Aggregation by Variable Objective Weighting from Hajela and Lin [33]; (iii) Niched Pareto Genetic Algorithm (NPGA) from Horn, Nafpliotis and Goldberg [35]; and (iv) Nondominated Sorting Genetic Algorithm (NSGA) from Srinivas and Deb [61]. The SPEA algorithm is char-

acterized by: (a) storing nondominated solutions externally in a second continuously updated population, (b) evaluating an individual's fitness dependent on the number of external non-dominated points that dominate it, (c) preserving population diversity using Pareto dominance relationship, and (d) incorporating a clustering procedure in order to reduce the nondominated set without destroying its characteristics [65].

The Pareto Archived Evolution Strategy (PAES) proposed by Knowles and Corne [42] is intended to be a simpler evolution scheme for multi-objective optimization problems. The PAES algorithm can be defined as a (1 + 1) evolution strategy, i.e., that a population of size two with one parent and its mution is used. However, they use a reference archive of previously find solutions in order to identify the non-domination ranking of the current and candidate solutions.

[15] proposes a modified approach of the NSGA-II algorithm. The NSGA-II algorithm has difficulties in solving problems with a large number of objectives, and the modified approach uses a reference point method to find Pareto optimal solutions near the DM's regions of interest. An interactive approach is described in which the obtained solution is used to create M new reference points. New Pareto optimal solutions are then found by forming new achievement scalarizing problems. A new reference point will be suggested if none of the suggested solutions satisfies the DM, and this procedure is repeated until the DM is satisfied. However, this procedure results to a single solution that depends on the chosen weight vector. Therefore, [15] proposes a reference point method based of the NSGA-II algorithm (R-NSGA-II) to obtain multiple solutions without the need of a (subjective) weight vector.

[10] describes an interactive approach, together with a genetic algorithm to solve the model with preferences included. During the optimization process of the genetic algorithm, the DM is asked to compare some pairs of solutions in the current population. Then a set of value functions is used together with the preference information and a modified version of the NSGA-II is applied to search solutions satisfying the DM's preferences.

Another interactive approach to include preference along with evolutionary optimization is described in [63]. In this paper they propose an algorithm called Preference-Based Evolutionary Algorithm (PBEA), based on the IBEA. The DM is asked to give his/her preference information in each information by defining a reference point. This algorithm discovers the Pareto front near the reference points given by the DM, and thus the whole Pareto front does not have to be approximated.

2.2.4 Implementation of GAs for supply chain design

To implement the supply chain in a genetic algorithm, you need a representation of the design and flows between the different sources. If the supply chain consists of multiple stages, each stage requires a part of the chromosome. Thus for each stage of the supply chain, a part of the chromosome is dedicated to represent from which source to which depot the goods are transported, how many goods there are transported and also the transportation modes if there are multiple available.

The optimization of a supply chain with a genetic algorithm is described in [1, 62]. In the first paper [1], the authors provide a genetic algorithm for solving a multi-objective supply chain network, with costs, customer service and capacity utilization ratio as the objectives. The

second [62] is an implementation for a single-objective optimization: the costs. Important here for our research is the representation and the genetic operators. As stated in the first paper, tree-based representation is one of the best ways for representing network problems, and both of these papers use a tree-based representation. For tree-based representations, there are three ways of encoding: edge, vertex or edge-and-vertex encoding. The second paper uses vertex-based encoding using Prüfer number representation [27], which needs a repair mechanism for infeasible solutions. In the first paper they use priority-based encoding to escape from the repair mechanism, which is edge-and-vertex encoding. Three genetic operators are applied in both papers: crossover, mutation and a selection mechanism.

As mentioned above, the representation of the network and flows can be encoded/decoded using priority-based encoding developed by Gen and Cheng [27]; [1] uses such encoding for a two-stage transportation problem. With priority-based encoding the length of a chromosome is equal to the number of sources plus the number of depots. If more stages are required, additional parts needed to be represented. This can be done in the same way as for the transportation from the first to the second stage.

The chromosomes have the length of the number of sources ($|K|$) plus the number of depots ($|J|$), i.e. $|K|+|J|$. Thus the chromosome (or part of the chromosome) consists of the priorities, where the first K integers represent the sources and the next J integers represent the depots. Then, based on the priorities, arcs are sequentially added between sources and depots. Figure 4 shows an example of this. Algorithm 3 presents the procedure to decode a chromosome for a single stage of the network [1]. It shows how a single stage can be decoded to the corresponding flows in the supply chain.. Because the length of the chromosome is $|K|+|J|$, it is guaranteed that each source or depot is served and thus the demand conditions are met (only if the capacity allows this and thus a feasible solution is available). However, these parts now represent arcs in the transportation tree, but not yet which transportation mode is used. Therefore, a part of equal size ($|K|+|J|$) can be added to the chromosome which consists of integer numbers to represent the different transportation modes. Thus, e.g., in case of three sources and four depots the chromosome consists of two parts (both of size ($|K|+|J|$)), where both parts have length seven. The first part represent the transportation modes and the second part the priorities. If you split these parts and link them, i.e. the first integer in the first part corresponds to the first integer in the second part, etc., you can combine a specific arc (based on the priority) with the transportation mode.

2.3 MOO in Green Logistics

In this research we investigate the impact of greening the supply chain on the design and cost. The case presented in Chapter 4, involves besides the minimization of the costs, also the minimization of CO_2/PM emissions. Therefore, we discuss (a part of) the related work about greening the supply chain.

The Locating of hazardous waste treatment and their routing has been studied with MOO models. In [36] is a goal programming model developed that considers four objectives: (i) minimization of total operation cost, (ii) minimization of total perceived risk, (iii) equitable

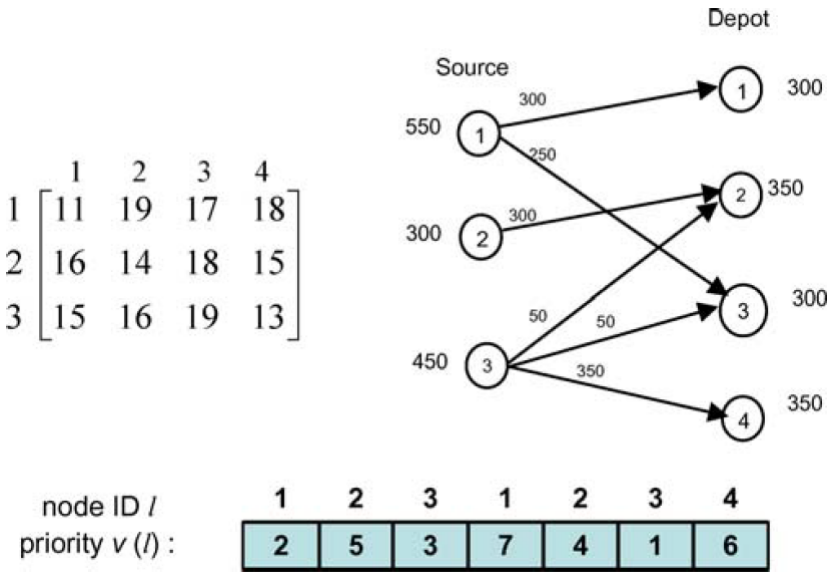


Figure 4: A sample of priority-based encoding; a transportation tree and its encoding [1].

Algorithm 3 Decoding the chromosome for transportation tree

Input:	K	:	set of sources, J : set of depots;
	b_j	:	demand of depots j, $\forall\, j \in J$;
	a_k	:	capacity of source k, $\forall\, kinK$;
	c_{kj}	:	transportation costs of one unit of product from source k to depot j, $\forall\, k \in K, \forall\, jinJ$;
	$v(k+j)$	:	chromosome, $\forall\, k \in K$, $\forall\, j \in J$;
Output:	g_{kj}	:	the amount of products shipped from source k to depot j;

Step 1: $g_{kj} \leftarrow 0, \forall\, k \in K, \forall\, j \in J$
Step 2: $l \leftarrow \arg\max\{v(t),\, t \in |K| + |J|\}$; select a node
Step 3:
  if $l \in K$ **then**
    $k^* \leftarrow l$; select a source
    $j \leftarrow \arg\min\{c_{kj} | v(j) \neq 0,\, j \in J\}$; select a depot with the lowest cost
  else
    $j^* \leftarrow l$; select a depot
    $k^* \leftarrow \arg\min\{c_{kj} | v(j) \neq 0,\, k \in K\}$; select a source with the lowest cost
Step 4: $g_{k^*j^*} \leftarrow \min\{a_{k^*}, b_{j^*}\}$; assign available amount of units
      Update availabilities on source (k^*) and depot (j^*)
        $a_{k^*} = a_{k^*} - g_{k^*j^*}$
Step 3:
  if $a_{k^*} = 0$ **then**
    $v(k^*) = 0$
  if $b_{j^*} = 0$ **then**
    $v(j^*) = 0$
Step 5:
  if $v(|K| + j) = 0, \forall\, j \in J$ **then**
    calculate transportation cost and return
  else
    goto Step 1

distribution of risk among population centers, and (iv) equitable distribution of the disutility caused by the operation of the treatment facilities. A bi-objective model for locating hazardous waste and routing has been proposed by [2], that include the optimization objectives of minimizing the total cost and the transportation risk. [50] is another study that integrate both risk and cost.

Quariguasi Frota Neto et al. [51] want to balance the cost and the environmental impact. They correctly argue that an improvement of sustainable logistics is often only possible with an investment that brings none or negative results. They present a framework for optimizing both cost and environmental impact simultaneously, consisting of three steps: (i) asses the environmental impact, (ii) normalization, and (iii) weighting. Then they want to explore all Pareto optimal solutions, using a heuristic to present the multi-objective optimization problem as several single-objective optimization problems. Harris et al. [34] evaluates a supply chain on overall logistics costs and CO_2 emissions by taking into account the structure (e.g. the number of depots) and different freight utilization ratios; the case study involve operational and strategic decisions.

Apart from multi-objective optimization in green logistics, there are also studies that focus just on the green logistics part. Sbihi and Eglese [55] focus on three topics: reverse logistics (recycling), waste management and vehicle routing/scheduling. For each of these topics different theories are applied, e.g., (facility) location models and dynamic lot-sizing for recycling and arc routing for waste management. The vehicle routing and scheduling consists of the link between vehicle routing and emissions. Sbihi and Eglese want to reduce the emissions by reducing the travel distance, i.e., route optimization. Zheng and Zhang [64] provide detailed steps for green logistics. They divide the green logistics into five parts: (a) green packaging, (b) green transport, (c) green storage, (d) green flow of processing, and (e) development of reverse logistics. For each of these parts there are a number of green aspects to apply, e.g., using greener packaging material or a greener way of transport. They split Reverse logistics into two parts: recycling and waste management. The first one is about reusing material, while the second one is about handling, storage, processing and distribution of waste.

Chapter 3

Preference modeling

The previous chapter describes the multi-objective optimization process, which results in a set of possible solutions. Often, it is difficult to choose one solution from a (large) set of solutions. However, it is much easier for a DM to make a choice between two alternatives. So-called preference models can find the most preferred solutions by applying piecewise comparisons in an iterative process. Thus for any pair of alternatives (x and y), the DM has to choose which one he/she prefers, and then the model induces the DM's most preferred solution from the (large) solutions' set. The crux of the preference model is to come up with such a sequence of piecewise comparisons, which should be as short as possible.

The classical theory of preference modeling can be characterized by: (a) the use of a specific language, (b) the use of a specific syntax, and (c) the emphasis put on a number of particular situations [9]. In this chapter we describe first the Multi-Attribute Value Theory (MAVT) and the UTA method, which form the basis for UTA$^{\text{GMS}}$. Then we describe UTA$^{\text{GMS}}$, which we will apply on a case study, and other approaches such as the ELECTRE and PROMETHEE methods.

3.1 Multi-Attribute Value Theory

The theory we discuss here is based on the outline of the method in [57] of the Multi-Attribute Value Theory (MAVT). A few axioms proposed by Keeney and Raiffa [39] are used to build up the MAVT. It is necessary to represent the preference of a decision maker (DM) via a set of scores to compare different solutions. Also, weights are used to quantify the relative importance of criteria and to account for the differences in scales of the criteria scores. Certain conditions are necessary for the use of additive MAVT models. To illustrate these conditions, consider a decision problem with a number of alternative solutions. Let two of these solutions A and B, measured against two sets of criteria I and J, where I contains at least two criteria, and J contains at least one. A and B can be expressed as vectors of attribute levels, i.e. $A = (a_i, a_j)$ and $B = (b_i, b_j)$. In defining the attributes on which to measure the solutions, one must ensure that mutual preferential independence exists among them. This means that for comparisons in which some of the criteria are kept fixed, preference is determined solely by the criteria in which there is a variation and does not depend on the levels of the other criteria. I is preferentially

independent of J if for all a_i, b_i, preferences on criteria I:

$$\text{for some } \alpha_j \in J, \quad (a_i, \alpha_j) \leq (b_i, \alpha_j) \Longrightarrow (a_i, \beta_j) \leq (b_i, \beta_j) \quad \forall \beta_j \in J$$

This condition implies that the DM should be able to establish a weak ordering of their preferences, i.e. the DM can decide whether a_i is at least as good as b_i on criterion i: $a_i \leq b_i$.

The MAVT consists of two steps: (i) build a value function v_j for each criterion, and (ii) compute a global value $v(a_i)$ for each action a_i

$$v(a_i) = f\big(v_1(a_i), v_2(a_i), ..., v_n(a_i)\big)$$

The value functions created in the first step need to comply with the transitivity conditions of preferences and indifferences. For the second step, often the additive model is used:

$$v(a_i) = \sum_{j=1}^{n} w_j v_j(a_i) = w_1 v_1(a_i) + w_2 v_2(a_i) + ... + w_n v_n(a_i),$$

usually with $\sum_{j=1}^{n} w_j = 1$, where w_j are the weights or scale coefficients (with $w_j > 0$).

3.2 UTA

The UTA (UTilitè Additive) method assesses a set of additive value functions which aggregate multiple criteria in a composite criterion, using weak-order prefence information from the DM and the multi-criteria evalutions of these solutions. The method, proposed in [37], assesses a set of value functions using an ordinal regression method (linear programming) instead of a single value function for each criterion. A weak preference relation $\succsim$ on A_R, i.e. for each pair of vectors $x, y \in G$ is defined as:

$$x \succsim y \Longleftrightarrow x \text{ is at least as good as } y$$

and can be decomposed into asymmetric and symmetric parts as follows:

1. $x \succ y \equiv [x \succsim y \text{ and not } (y \succsim x)] \Longleftrightarrow$ 'x is preferred to y';
2. $x \sim y \equiv [x \succsim y \text{ and } y \succsim x] \Longleftrightarrow$ 'x is indifferent to y'.

The value function is additive if it has the form:

$$g(a) = \sum_{i=1}^{n} g_i(a_i) \tag{3.1}$$

Taking into account the additive form (Eq. 3.1) and the preference conditions (a weak order), the value of each solution a (from a set of solutions A_R) can be written as:

$$u'[g(a)] = \sum_{i=1}^{n} u_i[g_i(a)] + \sigma(a) \quad \forall \alpha \in A_R \tag{3.2}$$

where $\sigma(a)$ is the potential error relative to $u'[v(a)]$.

[37] proposes to use linear interpolation in order to estimate the corresponding marginal value functions in a piecewise linear form. For each criterion (g), the interval $[g_{i^*}, g_i^*]$ is split into $(\alpha - 1)$ equal intervals, and the endpoints g_i^j are given with:

$$g_i^j = g_{i^*} + \frac{j-1}{\alpha_i - 1}\left(g_i^* - g_{i^*}\right) \quad \forall\, j = 1, 2, ..., \alpha_i \tag{3.3}$$

Then the marginal value of a is approximated by linear interpolation, and thus, for $g_i(a) \in [g_i^j, g_i^{j+1}]$:

$$u_i[g_i(a)] = u_i(g_i^j) + \frac{g_i(a) - g_i^j}{g_i^{j+1} - g_i^j}\left[u_i(g_i^{j+1}) - u_i(g_i^j)\right] \tag{3.4}$$

Combining Eq. 3.2 with the relations of a weak order, we have Eq. 3.5 for a strict preference relation (a is better than b: $a \succ b$) and Eq. 3.6 for indifferent pairs (a is equal to b: $a \sim b$).

$$\sum_{i=1}^{n}\left\{u_i[g_i(a)] - u_i[g_i(b)]\right\} + \sigma(a) - \sigma(b) > 0 \tag{3.5}$$

$$\sum_{i=1}^{n}\left\{u_i[g_i(a)] - u_i[g_i(b)]\right\} + \sigma(a) - \sigma(b) = 0 \tag{3.6}$$

Taking into account the hypothesis on monotonicity of preferences (i.e. the marginal values are monotone non-decreasing functions of the criterion g_i), the marginal values $u_i(g_i)$ must satisfy the constraint in Eq. 3.7, with s_i being the indifference threshold for each criterion g_i.

$$u_i(g_i^{j+1}) - u_i(g_i^j) \geq s_i, \qquad \forall\, j = 1, 2, ..., \alpha - 1, i = 1, 2, ..., n \tag{3.7}$$

The values $u_i(g^j)$ are estimated by means of a linear program where we use a linear objective function in order to minimize the amount of total deviation. The objective function:

$$\min F = \sum_{a \in A_R} \sigma(a) \tag{3.8}$$

Under the constraints:

$$\begin{aligned}
&\sum_{i=1}^{n}\left\{u_i[g_i(a)] - u_i[g_i(b)]\right\} + \sigma(a) - \sigma(b) > 0 && \text{if } a \succ b \\
&\sum_{i=1}^{n}\left\{u_i[g_i(a)] - u_i[g_i(b)]\right\} + \sigma(a) - \sigma(b) = 0 && \text{if } a \sim b \\
&u_t(g_i^{j+1}) - u_i(g_i^j) \leq s_i && \forall\, i, \forall\, j \\
&\sum_{i=1}^{n} u_i(g_i^*) = 1 \\
&u_i(g_{i^*}) = 0,\ u_i(g_i^j) \geq 0,\ \sigma(a) \geq 0 && \forall\, i, \forall\, j, \text{and} \forall a \in A_R
\end{aligned} \tag{3.9}$$

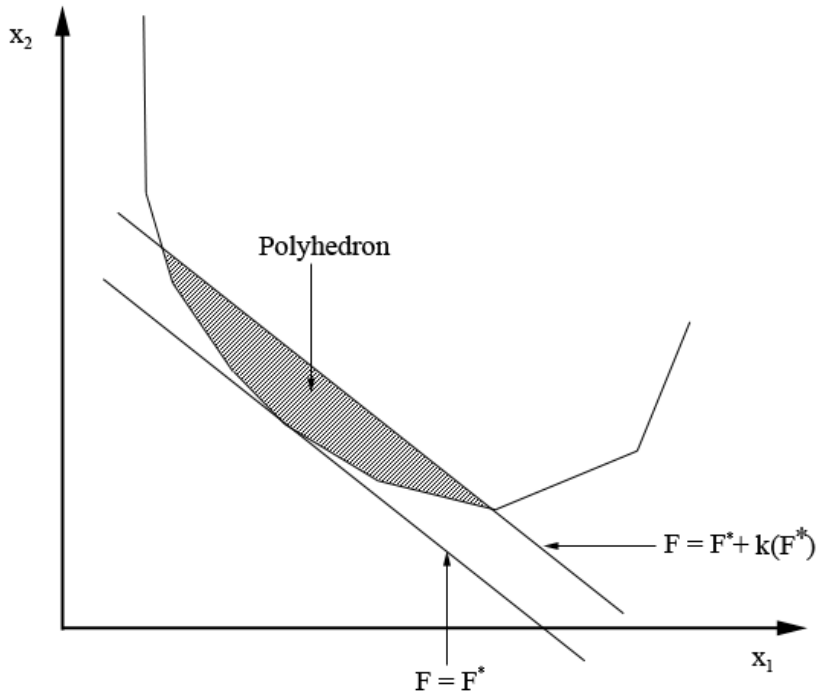


Figure 5: Post-optimality analysis [37].

The stability analysis of the results of the linear problem is considered as a post-optimality analysis problem. If the optimum F^* is zero, the polyhedron[1] of admissible solutions for $u_i(g_i)$ is not empty and many value functions lead to a perfect representation of the weak order R. Figure 5 shows the post-optimal solution space defined by the polyhedron:

$$F \leq F^* + k(F^*) \tag{3.10}$$

under the constraints of Eq. 3.9 where $k(F^*)$ is a positive threshold which is a small proportion of F^*. Branch and bound algorithms can be used to explore the polyhedron, and these results can give a very clear idea of the stability of $u_i(g_i)$. Partial exploration of the polyhedron can be obtained by solving the LPs $[\min]u_i(g_i^*)$ and $[\max]u_i(g_i^*)$ in the polyhedron (Eq. 3.10). In case of instability, the latter two LPs give the internal variation of the weights of criteria g_i, and consequently give an idea of the importance of the criteria to the DM.

UTASTAR

[60] describes the UTASTAR algorithm, which is an improvement of the UTA method by Siskos and Yannacopoulos [59]. The error function, with single error $\alpha(a)$, is not sufficient to minimize completely the dispersion of points all around the monotone curve of Figure 6. The points on the right side of the curve cause this problem; it would be suitable to subtract an amount of value and not increase the values of the others. Thus a double positive error function is proposed,

[1] A geometric solid in three dimensions with flat faces and straight edges

where σ^+ and σ^- are the overestimated and underestimated errors respectively:

$$u'[g(a)] = \sum_{i=1}^{n} u_i[g_i(a)] - \sigma^+(a) + \sigma^-(a) \quad \forall a \in A_R \tag{3.11}$$

Another important modification concerns the monotonicity of preference constraints of the criteria. Instead of an indifference threshold for each criterion, the threshold can be replaced by a non-negativity constraint for the variable w_{ij} (for $s_i = 0$):

$$w_{ij} = u_i(g_i^{j+1}) - u_i(g_i^j) \geq 0 \qquad \forall i = 1, 2, ..., n \text{ and } j = 1, 2, ..., \alpha_i - 1 \tag{3.12}$$

Consequently, the UTASTAR algorithm can be summarized in the following steps ([60]):

1. Express the global value of reference actions $u[g(a_k)], k = 1, 2, ...m$, first in terms of marginal values $u_i(g_i)$, and then in terms of variables w_{ij} according to the Equation 3.11, by means of the following expression:

$$\begin{cases} u_i(g_i^1) = 0 & \forall i = 1, 2, ..., n \\ u_i(g_i^j) = \sum_{t=1}^{j-1} w_{it} & \forall i = 1, 2, ..., n \text{ and } j = 2, 3, ..., \alpha_i - 1 \end{cases} \tag{3.13}$$

2. Introduce two error functions σ^+ and σ^- by writing for each pair of consecutive actions in the ranking analysis expressions:

$$\begin{aligned} \Delta(a_k, a_{k+1}) \quad = \quad & u[g(a_k)] - \sigma^+(a_k) + \sigma^-(a_k) \\ & -u[g(a_{k+1})] + \sigma^+(a_{k+1}) - \sigma^-(a_{k+1}) \end{aligned} \tag{3.14}$$

3. Solve the linear program:

$$\begin{aligned} & \min z = \sum_{k=1}^{m} [\sigma^+(a_k) + \sigma^-(a_k)] \\ & \text{subject to} \\ & \left.\begin{array}{ll} \Delta(a_k, a_{k+1} \geq \delta & \text{if } a_k \succ a_{k+1} \\ \Delta(a_k, a_{k+1} = 0 & \text{if } a_k \sim a_{k+1} \end{array}\right\} \quad \forall k \\ & \sum_{i=1}^{n} \sum_{j=1}^{\alpha_i - 1} w_{ij} = 1 \\ & w_{ij} \geq 0, \sigma^+(a_k) \geq 0, \sigma^-(a_k) \geq 0 \quad \forall i, j \text{ and } k \end{aligned} \tag{3.15}$$

   with δ being a small positive number.

4. Test the existence of multiple or near optimal solutions of the linear program (Eq. 3.15)(stability analysis); in case of non uniqueness, find the mean additive value functions of those (near) optimal solutions which maximize the objective functions:

$$u_i(g_i^*) = \sum_{j=1}^{\alpha_i - 1} w_{ij} \quad \forall i = 1, 2, .., n \tag{3.16}$$

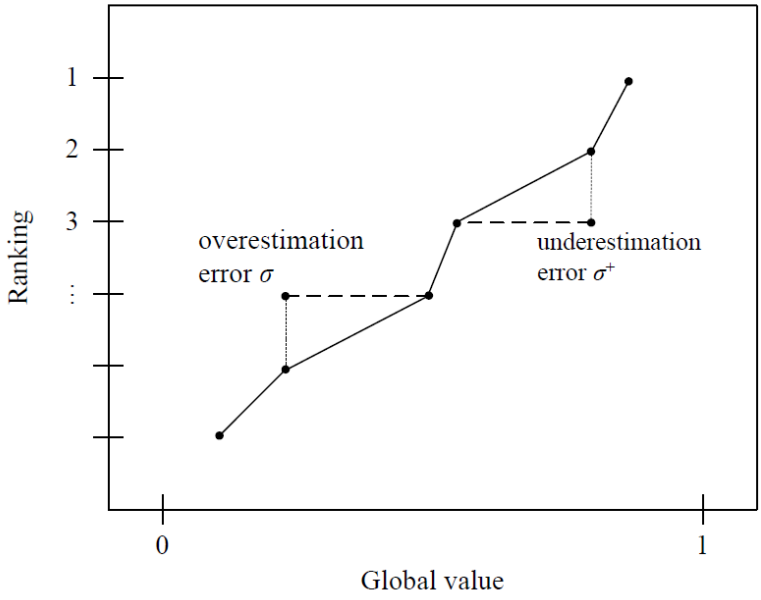


Figure 6: Ordinal regression curve; ranking versus global value [60].

on the polyhedron of constraints of the LP (3.15) bounded by the new constraint:

$$\sum_{k=1}^{m} \left[\sigma^{+}(a_k) + \sigma^{-}(a_k)\right] \leq z^{*} + \varepsilon \tag{3.17}$$

where z^{*} is the optimal value of the LP in step 3 and ε a very small positive number.

The comparison between UTA and UTASTAR made by [59] provided that UTASTAR has better results concerning a number of comparison indicators, like:

1. The number of the necessary simplex iterations for arriving the optimal solutions;
2. Kendall's $\boldsymbol{\tau}$ [2] between the initial weak order and the one produced by the estimated model;
3. The minimized criterion z (sum of errors) taken as the indicator of dispersion of the observation.

3.3 UTA$^{\mathbf{GMS}}$

The UTA$^{\mathrm{GMS}}$ method is proposed by Greco et al. [29] and can be outlined as follows. The preference information in this method is in form of pairwise comparisons of reference alternatives $A^R \subseteq A$ and is a partial preorder on A^R, denoted by $\succsim$. A value function is compatible if it is able to restore the partial preorder $\succsim$ on A^R. Furthermore, each compatible value function induces a complete preorder on the whole set A.

In particular, any two alternatives $x, y \in A$, can be ranked in the following way by a compatible value function: $x \succ y, y \succ x$, and $x \sim y$. With respect to $x, y \in A$, it is thus reasonable to ask the following two questions:

[2] Correlation coefficient based on ranking instead of the data itself

1. are x and y ranked in the same way by all compatible value functions?

2. is there at least one compatible value function ranking x at least as good as y (or y at least as good as x)?

In answering these questions for all pairs of alternatives $(x, y) \in A \times A$, one gets a necessary weak preference relation $\succsim^N$ in A ($U(x) \geq U(y)$ for all compatible value functions), and a possible weak preference relation $\succsim^P$ in A ($U(x) \geq U(y)$ for at least one compatible value function).

In order to compute the binary relations $\succsim^N$ and $\succsim^P$, UTA$^{\text{GMS}}$ continues the following way. For all actions $x, y \in A$, let π_i be a permutation of indices of alternatives from set $A^R \cup x, y$ that reorders them increasingly on criterion g_i, i.e. $g_i(a_{\pi_i(1)}) \leq g_i(a_{\pi_i(2)}) \leq ... \leq g_i(a_{\pi_i(\omega)})$, where

- if $A^R \cap \{x, y\} = \emptyset$, then $\omega = m + 2$;
- if $A^R \cap \{x, y\} = x$ or $A^R \cap \{x, y\} = y$, then $\omega = m + 1$;
- if $A^R \cap \{x, y\} = \{x, y\}$, then $\omega = m$.

Then we can fix the characteristic points of $u_i(g_i)$, $i = 1, 2, ..., n$, in

$$g_i^0 = \alpha_i,\ g_i^j = g_i(a_{\pi_i(j)}), \quad \text{for } j = 1, 2, ..., \omega \quad , g_i^{\omega+1} = \beta_i$$

Let us consider the following ordinal regression constraints:

$$\begin{array}{l} \left.\begin{array}{l} U(a) \geq U(b) + \varepsilon \Longleftrightarrow a \succ b \\ U(a) = U(b) \Longleftrightarrow a \sim b \end{array}\right\} \quad \forall a, b \in A^R \\ u_i(g_i^j) - u_i(g_i^{j-1}) \geq 0, \quad i = 1, 2, ..., n, \quad j = 1, ..., \omega + 1 \\ u_i(g_i^0) = 0, \quad i = 1, 2, ..., n \\ \sum_i^n (g_i^{\omega+1}) = 1 \end{array} \tag{3.18}$$

where ε is an arbitrary small positive value. This set of constraints depends on the pair of alternatives $x, y \in A$, because $g_i(x)$ and $g_i(y)$ give coordinates for two of $(\omega + 1)$ characteristic points of marginal value function $u_i(x_i) \quad \forall i$.

We suppose that the polyhedron defined by the set of constraints (3.18) is not empty. Thus in this case we have:

$$x \succsim^N y \Longleftrightarrow d(x, y) \geq 0$$

where

$$\begin{array}{rll} d(x, y) = & \min & \{U(x) - U(y)\} \\ & \text{s.t.} & \text{(Eq. 3.18)} \end{array}$$

and

$$x \succsim^P y \Longleftrightarrow D(x, y) \geq 0,$$

where

$$D(x,y) = \begin{array}{ll} \max & \{U(x) - U(y)\} \\ \text{s.t.} & \text{(Eq. 3.18)} \end{array}$$

3.4 UTA variants and extensions

3.4.1 GRIP

Generalized Regression with Intensities of Preference (GRIP) generalizes both UTA (UTASTAR actually) and UTA$^{\text{GMS}}$ methods by adopting all features of UTA$^{\text{GMS}}$ and taking into account additional preference information in the form of comparisons of intensities of preferences. This method is proposed by Figueira et al. [25] and can be outlined as follows. For alternatives $x, y, w, z \in A$, the comparisons can be expressed in two possible, not exclusive, ways:

1. Comprehensively, on all criteria, like 'x is preferred to y at least as much as w is preferred to z';
2. Partially, on a particular criteria, like 'x is preferred to y at least as much as w is preferred to z', on criterion $g_i \in G$.

Additional preference information needed to provide by the DM, besides the partial preorder $\succsim$ on A^R, consists of:

- A partial preorder $\succsim^*$ on $A^R \times A^R$, whose meaning is: for $x, y, w, z \in A^R$:

$$(x,y) \succsim^* (w,z) \iff \text{'}x \text{ is preferred to } y \text{ at least as much as } w \text{ is preferred to } z\text{'}$$

- A partial preorder $\succsim_i^*$ on $A^R \times A^R$, whose meaning is: for $x, y, w, z \in A^R$:

$$(x,y) \succsim_i^* (w,z) \iff$$
$$\text{'}x \text{ is preferred to } y \text{ at least as much as } w \text{ is preferred to } z\text{' on criterion } g_i, i \in I$$

Also is stated that intensities of preferences can be expressed in terms of pre-defined degrees of intensities such as 'moderate' or 'very strong', e.g., the preference of x over y is moderate, the preference of w over z is very strong.

The following output can/is produced by the GRIP method:

- a necessary ranking $\succsim^N$, for all pairs of actions $(x,y) \in A \times A$;
- a possible ranking $\succsim^P$, for all pairs of actions $(x,y) \in A \times A$;
- a necessary ranking $\succsim^{*^N}$, with respect to the comprehensive intensities of preferences for all $\big((x,y),(w,z)\big) \in A \times A \times A \times A$;
- a possible ranking $\succsim^{*^P}$, with respect to the comprehensive intensities of preferences for all $\big((x,y),(w,z)\big) \in A \times A \times A \times A$;

- a necessary ranking $\succsim_i^{*N}$, with respect to the comprehensive intensities of preferences for all $\big((x,y),(w,z)\big) \in A \times A \times A \times A$ and for all criteria g_i, $i \in I$;
- a possible ranking $\succsim_i^{*P}$, with respect to the comprehensive intensities of preferences for all $\big((x,y),(w,z)\big) \in A \times A \times A \times A$ and for all criteria g_i, $i \in I$;

However, the first two (the necessary $\succsim^N$ and possible $\succsim^P$ ranking) provide the most useful information and there is often no actual need to compute the other results, they can be requested concerning particular pairs of alternatives.

3.4.2 Stochastic UTA

The stochastic UTA method is proposed by Siskos [58] for multi-criteria decision aid under uncertainty. The additive value function becomes:

$$u(\delta^a) = \sum_{i=1}^{n} \sum_{j=1}^{a_i} \delta_i^a(g_i^j) u_i(g_i^j) \tag{3.19}$$

where δ_i^a is the distributional evaluation of alternative a on criterion i, $\delta_i^a(g_i^j)$ is probability that the performance of alternative a on criterion i is g_i^j, and $u_i(g_i^j)$ is the marginal value of performance g_i^j.

3.4.3 UTADIS

The UTA-method can be extended to use for sorting problems, e.g., with discriminant analysis. In case of two classes, the following two inequalities must hold:

$$\begin{cases} a \in A_1 \Leftrightarrow u[g(a)] \geq u_0 \\ a \in A_2 \Leftrightarrow u[g(a)] < u_0 \end{cases} \tag{3.20}$$

with u_0 being the level of acceptance/rejection in order to distinguish alternatives being in A_1 or A_2, where A_1 contains the most preferred alternatives. Instead of two classes, there can be many classes. Typically, the DM evaluation is expressed in terms of classification of reference groups $A_1 \succ A_2 \succ \ldots \succ A_q$. Then the following conditions have to be satisfied:

$$\begin{cases} u[g(a)] > u_i & \forall a \in A_1 \\ u_l \leq u[g(a)] < u_{l-1} & \forall a \in A_l \, (l = 2, 3, \ldots, q-1) \\ u[g(a)] < u_{q-1} & \forall a \in A_q \end{cases} \tag{3.21}$$

where $u_1 < u_2 < \ldots < u_{q-1}$ are thresholds to discriminate the groups. Similar to UTASTAR, two error variables are used (σ^+ and σ^-). Then the additive value model is developed to minimize these errors. Recently, several new variants are published such as UTADIS I, II and III.

3.4.4 UTADIS$^{\mathbf{GMS}}$

Greco et al. [30] proposed the UTADIS$^{\text{GMS}}$ method, which is an ordinal regression method for multiple-criteria sorting problems using a set of additive value functions as preference model. Like other robust ordinal regression methods, it takes into account the set of all value functions compatible with the DM's preferences. Furthermore, it considers general monotone increasing marginal value functions instead of piecewise linear only (like UTADIS).

The DM is asked to provide a set of assignment examples. Each assignment example consists of an alternative and its desired assignment, i.e., a lower bound and upper bound for its desired class. Then considering all compatible value functions, two possible assignments are obtained for each alternative a: the possible and the necessary assignment. The necessary assignment $C^N_{d_r}(a)$ determines the classes A_h for which all compatible value functions assign a to A_h, and the possible assignment $C^P_{d_r}$ determines the classes C_h for which exists at least one compatible value function that assigns a to C_h.

The procedure consists of six steps, of which the first three initial steps are the same as with UTA$^{\text{GMS}}$. The remaining steps involve the calculation of the boundary indices and the necessary and possible classes and their assignments.

3.4.5 UTA$^{\mathbf{GMS}}$-GROUP and UTADIS$^{\mathbf{GMS}}$-GROUP

Robust ordinal regression for group decision problems is proposed in [32]; both the UTA$^{\text{GMS}}$-GROUP and UTADIS$^{\text{GMS}}$-GROUP are introduced and extend the original method to group decision making. In the group context, let us denote a set of DM with $\mathcal{D} = \{d_1, d_2, \ldots, d_p\}$, and each member having an equal weight because it is assumed that they have the same importance. In the first stage, each individual member of $\mathcal{D}$ is asked to give his/her preferences, and then the possible and necessary relations/assignments are identified. In the second stage, spaces of consensus are investigated for subsets of DM. Again, the possible and necessary results verify whether these results are given by at least one DM or all DMs. This way they can identify what would happen always, sometimes or never. There are four possible types of results:

- Necessary–Necessary (N, N), the necessary (N) consequences provided by each DM are confirmed for all DMs (N);
- Necessary–Possible (N, P), the necessary (N) consequences provided by each DM are confirmed for at least one DM (P);
- Possible–Necessary (P, N), the results formed by the possible (P) outcomes are confirmed for all DMs (N);
- Possible–Possible (P, P), the results formed by the possible (P) outcomes are confirmed for at least one DM (P).

The type of result indicates the certainty of a possible action. When it is (N, N) it is absolutely sure, for (N, P) or (P, N) it can differ from absolutely sure to not sure at all sure, and the type (P, P) has the lowest level of certainty. However, if (P, P) is false, this negative result can be confirmed with the greatest confidence, because it is confirmed for all compatible models.

In case of UTA$^{\text{GMS}}$-GROUP (for ranking and choice problems), each DM is asked to give the preference information required by the UTA$^{\text{GMS}}$ and GRIP method. Using this preference information, the possible $\succeq_{d_r}^{P}$ and necessary $\succeq_{d_r}^{N}$ relation are computed for all DM $d_r \in \mathcal{D}$. Then the four possible types of preference relations can be determined for all subsets $\mathcal{D}' \subseteq \mathcal{D}$:

- $a \succeq_{\mathcal{D}}^{N,N} b : a \succeq_{d_r}^{N} b$ for all $d_r \in \mathcal{D}'$
- $a \succeq_{\mathcal{D}}^{N,P} b : a \succeq_{d_r}^{N} b$ for at least one $d_r \in \mathcal{D}'$
- $a \succeq_{\mathcal{D}}^{P,N} b : a \succeq_{d_r}^{P} b$ for all $d_r \in \mathcal{D}'$
- $a \succeq_{\mathcal{D}}^{P,P} b : a \succeq_{d_r}^{P} b$ for at least one $d_r \in \mathcal{D}'$

Then from these four relations, one can obtain indifference ($\sim$), preference ($\succ$), and incomparability (?) in a usual way.

In case of UTADIS$^{\text{GMS}}$-GROUP (for sorting problems), a set of all compatible value functions $\mathcal{U}_{A^R,d_r}$ is considered. Given a set $A_{d_r}^R$ of assignment examples, for each $a \in A$ and for each $d_r \in \mathcal{D}$, the possible and necessary assignment are defined like in UTADIS$^{\text{GMS}}$:

$$C_{d_r}^{P}(a) = \left\{h \in H : \exists \mathcal{U} \in \mathcal{U}_{A^R,d_r} \text{assigning } a \text{ to } C_h\right\} \text{ and}$$
$$C_{d_r}^{N}(a) = \left\{h \in H : \forall \mathcal{U} \in \mathcal{U}_{A^R,d_r} \text{assigning } a \text{ to } C_h\right\}$$

Then the four types of assignments can be computed:

- $C_{\mathcal{D}}^{N,N}(a) = \cap_{d_r \in \mathcal{D}'} C_{d_r}^{N}(a)$,
- $C_{\mathcal{D}}^{N,P}(a) = \cup_{d_r \in \mathcal{D}'} C_{d_r}^{N}(a)$,
- $C_{\mathcal{D}}^{P,N}(a) = \cap_{d_r \in \mathcal{D}'} C_{d_r}^{P}(a)$,
- $C_{\mathcal{D}}^{P,P}(a) = \cup_{d_r \in \mathcal{D}'} C_{d_r}^{P}(a)$.

In UTADIS$^{\text{GMS}}$ each set contains a set of value functions; these sets are ordered in the same order as the exemplary assignments. For each iteration, the possible $C_{t,d_r}^{P}(a)$ and necessary $C_{t,d_r}^{N}(a)$ assignments are computed. Then for all $a \in A$, we can compute the assignments $C_{t,\mathcal{D}}^{N,N}(a), C_{t,\mathcal{D}}^{N,P}(a), C_{t,\mathcal{D}}^{P,N}(a)$ and $C_{t,\mathcal{D}}^{P,P}$ (a).

3.5 Outranking methods

According to [57], an outranking relation is a binary relation which compares the arguments for and against a hypothesis *Alternative A is at least as good as Alternative B* given what is known about a DM's preferences. The alternatives are then compared via the outranking relation and placed in a partial order. There is no reason for an outranking relation to be transitive nor complete [53].

The outranking relation, usually denoted by S, was proposed by Roy whose aim was to represent four realistic situations of preference: indifference, weak preference, strict preference and incomparability. One, two or three of these situations assigned to any pair of alternatives could be used to develop a satisfactory preference model. If a outranks b, there are sufficient

arguments that a is not worse than b (concordance test) and no essential reasons to refuse that a outranks b (non-discordance test).

3.5.1 ELECTRE methods

The first appearance of a method of the ELECTRE family was in the mid-sixties; ELECTRE I was proposed by Roy. Nowadays, there are several methods which belong to this family, we describe the main futures of the ELECTRE Methods below, mainly based on Figueira et al. [23].

According to Figueira et al., the ELECTRE methods are relevant in the following context:

1. The DM wants to include at least three criteria in the model;
2. Alternatives are evaluated (for at least one criterion) on an ordinal scale or on a interval scale;
3. A strong heterogeneity related with the nature of evaluations exists among criteria;
4. Compensation of the loss on a given criterion by a gain on another one may not be acceptable for the DM;
5. For at least one criterion the following holds true: small differences of evaluations are not significant in terms of preferences, while the accumulation of several small differences may become significant. The latter requires the introduction of discrimination thresholds which leads to a preference structure with a comprehensive intransitive indifference binary relation.

In the ELECTRE Methods the preferences are modelled using binary outranking relation S. Considering two alternatives x and y, there are four possible situations:

- aSb and not bSa, i.e. $a \succ b$ (a is strictly preferred to b);
- bSa and not aSb, i.e. $b \succ a$ (b is strictly preferred to a);
- aSb and bSa, i.e. $a \sim b$ (a is indifferent to b);
- Not aSb and not bSa, i.e. aRb (a is incomparable to b).

ELECTRE methods build one or several outranking relations. Notice that a new preference relation, R (incomparability), is introduced. This is useful when the DM is not able to compare two alternatives. The construction of an outranking relation is based on two major concepts and these needs to be fulfilled for validating the assertion aSb:

Concordance For an outranking aSb to be validated, a sufficient majority of criteria should be in favor of this assertion;

Non-discordance When the concordance condition holds, none of the criteria in the minority should oppose too strongly to the assertion aSb.

ELECTRE methods consists of two main procedures: the construction of one or several outranking relation(s) followed by an exploitation procedure. In the first, each pair of alternatives are compared in a comprehensive way. The latter is used to elaborate recommendations from the results obtained in the first phase. The recommendations depends on the problematic (choosing, ranking or sorting), and thus each method can be characterized by its construction and its exploitation procedures.

ELECTRE I's objective is to aid the DM to find a single (best) solution, and thus belongs to the choice problematic. This method is very simple and should only be applied when all the criteria have been coded in numerical scales with identical ranges. The concordance and non-discordance conditions should hold, so we calculate the concordance index as follows (assuming $\sum_{j \in J} w_j = 1$):

$$c(aSb) = \sum_{\{j: g_j(a) \geq g_j(b)\}} w_j$$

where $\{j : g_j(a) \geq g_j(b)\}$ is the set of indices for all criteria belong to the concordant coalition with the outranking relation aSb. The index must be greater than or equal to a given concordance level s, i.e., $c(aSb) \geq s$. The discordance holds if it is smaller than or equal to a given level v, i.e., $d(aSb) \leq v$. The discordance can be calculated as follows:

$$d(aSb) = \max_{\{j: g_j(a) < g_j(b)\}} \left\{ g_j(b) - g_j(a) \right\}$$

Both conditions have to be computed for every pair of alternatives (a, b) in the set A, where $a \neq b$. This procedure leads to a binary relation in comprehensive terms and thus to one of the four situations mentioned earlier.

The exploiting of the outranking relation (the second procedure) is used to identify a small as possible subset of alternatives. Such a subset, $\hat{A}$, may be determined with the help of the *graph kernel concept*, K_G [54]. If the graph contain direct cycles, a preprocessing step must take place where maximal direct cycles are reduced to singleton elements, forming thus a partition on A. Let $\bar{A}$ denote that partition, each class on $\bar{A} = \{\bar{A}_1, \bar{A}_2, ...\}$ is now composed of a set of equivalent alternatives. The preference relation defined on $\bar{A}$ is:

$$\bar{A}_p \succ \bar{A}_q \Longleftrightarrow \exists a \in \bar{A}_p \text{ and } \exists b \in \bar{A}_q \text{ such that } aSb \text{ for } \bar{A}) p \neq \bar{A}_q$$

All actions that form a cycle are considered indifferent. The interested reader can find more about ELECTRE methods in [23].

3.5.2 PROMETHEE methods

In this section we describe an overview of the PROMETHEE methods, based on [11]. The first two methods, PROMETHEE I (partial ranking) and PROMETHEE II (complete ranking) are proposed and presented for the first time in 1982 by J.P. Brans. A few years later, Brans and Mareschal developed PROMETHEE III (ranking based on intervals) and PROMETHEE IV (continuous case). They also presented a visual interactive module GAIA, which provide a graph-

ical representation supporting the PROMETHEE methodology. Furthermore, PROMETHEE V (MCDA including segmentation constraints) and PROMETHEE VI (representation of the human brain) are also presented by Brans and Mareschal.

The preference information required by the PROMETHEE methods consists of information between criteria and information within each criteria. Information between criteria is the relative importance of each criteria. These weights can be normalized such that $\sum_{j=1}^{k} w_j = 1$. The structure of the preference information for the information within criteria is based on pairwise comparisons. The deviation between the alternatives are considered for each criterion and the larger the deviation, the larger the preference. These preferences can be real numbers between 0 and 1, and holds zero if the deviation is negative, i.e. $p(a,b) > 0 \Rightarrow p(b,a) = 0$. For each criterion we can define a preference function:

$$\pi_j(a,b) = F_j(g_j(a) - g_j(b)) \tag{3.22}$$

There are six possible types of preference functions proposed in order to facilitate the identification. For each function, 0, 1 or 2 parameters have to be defined: q is a threshold of indifference, p is a threshold of strict preference and s is an interval between q and p. In order to construct a partial or complete ranking, first aggregate preference indices and outranking flows have to be defined. Let $a, b \in A$, and let:

$$\begin{cases} \pi(a,b) = \sum_{j=1}^{k} \pi_j(a,b) w_j, \\ \pi(b,a) = \sum_{j=1}^{k} \pi_j(b,a) w_j. \end{cases} \tag{3.23}$$

where $\pi(a,b)$ is the degree which a is preferred to b over all criteria and $\pi(b,a)$ how b is preferred to a. When $\pi(a,b)$ and $\pi(b,a)$ for each pair of alternatives are computed, a complete valued outranking graph can be obtained with arcs between all pairs of nodes.

The positive outranking flow expresses how alternative a outranks all other $(n-1)$ alternatives and the negative outranking flow expresses how alternative a is outranked by all other $(n-1)$ alternatives. The positive outranking flow:

$$\phi^+(a) = \frac{1}{n-1} \sum_{x \in A} \pi(a,x), \tag{3.24}$$

and the negative outranking flow:

$$\phi^-(a) = \frac{1}{n-1} \sum_{x \in A} \pi(x,a). \tag{3.25}$$

The higher $\phi^+(a)$, the better alternative a is, the lower $\phi^-(a)$, the better the other alternative. The partial ranking (PROMETHEE I) can be obtained from the positive and the negative outranking flows. Because both flows usually do not induce the same ranking, the intersection

is used. For the complete ranking (PROMETHEE II), the *net outranking flow* is computed:

$$\phi(a) = \phi^+(a) - \phi^-(a). \tag{3.26}$$

The higher net flow, the better the alternative.

3.6 Outranking with Robust Ordinal Regression

The principle of robust ordinal regression for outranking models has been applied to the ELECTRE methods and the PROMETHEE methods. We describe both below.

3.6.1 ELECTRE$^{\text{GKMS}}$

ELECTRE$^{\text{GKMS}}$ implements robust ordinal regression to outranking methods and is proposed in [31]. It is motivated by missing arguments in ELECTRE-like methods, and these are analogical to issues listed by the UTA$^{\text{GMS}}$ method [29]. In ELECTRE$^{\text{GKMS}}$, the whole set of outranking models consistent with the preferences of the DM are assessed. More specifically, the sets of compatible intra-criterion and inter-criteria model parameters are considered. The former can be supplied in a direct way with allowed values or indirect by pairwise comparisons of some reference alternatives per criterion. The latter, important coefficients, veto thresholds, and concordance thresholds are much more difficult to supply. Therefore, these values are inferred from pairwise comparisons stating the truth or falsity of an outranking relation.

Often the functions of the marginal concordance indices are limited to piecewise linear functions. Some methods extend the set of functions, e.g. PROMETHEE allows to choose from some predefined shapes. ELECTRE$^{\text{GKMS}}$ extends this further with $\psi_j(a,b), j = 1, ..., n$, which are monotone non-decreasing functions with respect to $g_j(a) - g_j(b)$, such that $\psi_j = k_j$ if a outranks b on criterion j, and $\psi_j = 0$ if b is strictly preferred over a on criterion j.

With respect to all compatible outranking relations, it is reasonable for one to ask two questions for every pair of alternatives $(a, b) \in A \times A$: is a at least as good as b for all compatible outranking models, and does there exist at least one compatible outranking model for which a is at least as good as b. These two questions result in the following two outranking relations:

- *a necessarily outranks b* ($aS^N b$), if a outranks b for all compatible models;
- *a possibly outranks b* ($aS^P b$), if a outranks b for at least one compatible model.

3.6.2 PROMETHEE$^{\text{GKS}}$

The scheme of robust ordinal regression can be adjusted to PROMETHEE I or II and is proposed by Kadziński et al. [38]. The pairwise comparisons stating the truth or falsity of an outranking relation are either provided at the level of construction or at the level of exploitation. The former refers to the strength of arguments in favor of a over b or the other way around. The latter refers to their final ranking position. Furthermore, the set of all compatible PROMETHEE preference models considered. Consequently, Kadziński et al. define and build general monotone preference functions, and use the aggregate preference indices and outranking flows scores.

The preference information required from the DM consists of pairwise comparisons of some reference alternatives $a \in A^R \subseteq A$. They distinguish two kinds of such relations, relations at the construction level and relations at the exploitation level. This is done because it is possible within the family of outranking methods that a outranks b, but b takes a higher rank in the final ranking. However, this distinction between the construction and exploitation level may not be visible for the DM, because it is assumed that the strengths of arguments in favor of a over b also means that a is at least as good ranked as b in their final rank. Other information that should be given by the decision maker is the intra-criterion preference information concerning the indifference and preference thresholds $p_j \geq q_j \geq 0, j = 1, ..., m$. It is allowed that the DM provides real intervals rather than precise values.

Compatible outranking models are composed of the sets of preference indices $\pi(a, b)$, marginal preference indices $\pi_j(a, b)$, outranking flow scores $\psi(a), \psi^+(a)$, and $\psi^-(a)$, indifference q_i, and preference p_j thresholds, satisfying a set of conditions. Then both the necessary and possible relations are computed considering a LP problem (see [38] for the conditions and the LP problems).

3.7 Other methods

3.7.1 AHP

The analytical hierarchy process (AHP) can also be used when the DM is facing multiple objectives [see 28, chap. 16]. AHP consists of five stages:

Stage 1: Set up the decision hierarchy.

Stage 2: Make pairwise comparisons.

Stage 3: Transform the comparisons into weights and check the consistency of the decision maker's comparisons.

Stage 4: Use the weights to obtain scores for the different options and make a provisional decision.

Stage 5: Perform sensitivity analysis.

In the first stage, a hierarchical tree is made where each attribute will be broken down into more details for each level in the tree. Below each of the lowest-level attributes in the tree are the alternatives added to the tree. In the second stage, for each level of the tree we need to do pairwise comparisons for each possible pair on a scale from 1 to 9; these scales are: (1) equally important, (3) weakly more important, (5) strongly more important, (7) very strongly more important, (9) extremely more important, or intermediate decisions (2, 4, 6, and 8). The third stage first normalizes the weights and then uses a mathematical approach based on eigenvalues. Furthermore, there is an inconsistency index which is designed to alert the DM for any inconsistency; index zero indicates no inconsistencies. In the fourth stage, for each path from the top of the hierarchy down to the lowest level representing the alternative, the scores are multiplied, and the results of the different paths summed leading to a final score for this particular alternative. The final stage consists of a sensitivity analysis to examine how sensitive the preferred course of action is to changes of judgment made by the DM.

However, AHP is not a valid methodology. Barzilai described that AHP is based on mathe-

matical errors [6]. The AHP errors that are listed include that the coefficients of linear preference functions cannot correspond to weights representating relative importance, the eigenvector method is not the correct method for constructing preference scales, the assignment of number 1-9 is arbitrary, and there is no foundation for these scales. Therefore, AHP should not be used for preference modeling.

3.7.2 MACBETH

MACBETH (measuring attractiveness by a categorical-based evaluation technique) is an interactive approach for quantifying value judgments about the elements of a finite set [5]. The technical component of MACBETH is a chain of four linear programs ($Mc1$ to $Mc4$), first proposed by Bana e Costa and Vansnick [4]. In the first step the DM is asked to do pairwise comparisons for any two actions a and b, by choosing one of the six categories:

(C_1) *very weak* difference of attractiveness;
(C_2) *weak* difference of attractiveness;
(C_3) *moderate* difference of attractiveness;
(C_4) *strong* difference of attractiveness;
(C_5) *very strong* difference of attractiveness;
(C_6) *extreme* difference of attractiveness.

During the questioning process a matrix is filled with the categorical judgments, consistency of a ranking is made and discusses with the DM until the transitivity property holds. Then by applying the preference information to the four linear programs, one of the two possible cases according to the optimal value c_{min} of $Mc1$ can result. In the first case, when $c_{min} = 0$, it is possible to associate a real number $v(a)$ with each element $a \in A$ in such a way that the following two measurement rules are satisfied:

1. $\forall\, a, b \in A : a \succ b \iff v(a) > v(b)$; i.e., the number assigned to alternative a is strictly greater than the number assigned to alternative b, if and only if, for the DM a is more attractive than b;

2. $\forall\, k, k' \in \{1, 2, 3, 4, 5, 6\}$ with $k \neq k'$ and $\forall\, a, b, c, d \in A$ with $(a, b) \in C_k$ and $(c, d) \in C_{k'}$: $k > k' \iff v(a) - v(b) > v(c) - v(d)$; i.e. that the differences in the numbers between a and b are strictly greater than between c and d, not belonging to the same category (this information is not asked to the DM, but follows indirectly from the matrix of judgments).

In this case, $MC2$ gives real numbers $v(a)(a \in A)$ to be associated with the alternatives in A. These numbers can then be plotted to verify whether the relative differences between alternatives is what the DM wants.

In the second case ($c_{min} > 0$), it is not possible to represent the preferences numerically. $MC2$ suggest values $v(a)(a \in A)$ that tries to reconcile the DM's initial inconsistent judgments. Another possibility for the DM is to revise his/her preferences, in this case the programs $Mc1$ and $Mc2$ can run again. An interactive learning process can be facilitated by $Mc3$ and $Mc4$, they can suggest some concrete modifications of the DM's judgment.

3.7.3 Robust ordinal regression in non-additive form

In previously described methods with ordinal regression, such as UTA$^{\text{GMS}}$, ELECTRE$^{\text{GKMS}}$ and PROMETHEE$^{\text{GKS}}$, the value functions have an additive form. It is, however, also possible to develop a preference model with non-additive robust ordinal regression. In [3] they describe how the value can be evaluated in terms of the Choquet integral, which allows to represent the interaction among criteria modelled by fuzzy measures. The DM is asked to make pairwise comparisons to express his/her preferences, the intensity of preferences of pairs of alternatives (like GRIP), and to supply pairwise comparisons on the importance of criteria. The model then defines a set of fuzzy measures such that the corresponding Choquet integral is compatible with the DM's preferences. Furthermore, the model uses linear programming to establish the possible and necessary relations.

3.8 Overview

The previous described methods are all most common preference models that can be applied to different circumstances. Two of them are for groups decisions, while the UTADIS methods should be used for sorting/classification problems. Most of the methods are used for choosing or ranking problems.

The UTA method assesses a set of linear value functions, compatible for all criteria, from piecewise comparisons given by the DM. The methods based on the outranking relation, like ELECTRE and PROMETHEE methods, require additional information such as important coefficients and veto thresholds, while AHP and MACBETH require to assign scales to the pairwise comparisons. Therefore, from the DM's perspective is the UTA method the easiest to apply, because he/she only has to provide preference information in the form of pairwise comparisons.

The principles of robust ordinal regression are applied to the UTA, ELECTRE and PROMETHEE methods. Robust ordinal regressions suits the preference models for cases where the DM's preferences cannot be presented by piecewise linear functions only. Robust ordinal regression takes into account the set of all compatible additive value functions, and considers non-decreasing marginal value functions. Also, robust ordinal regression is used to infer the inter-criteria information (i.e. important coefficients and veto thresholds) for the outranking models. The methods then compute the weak necessary and weak possible relations for a set of actions.

The GRIP method extends the UTA$^{\text{GMS}}$ methods by taking additional preference information into account in the form of comparisons of intensities of preferences. This additional information is useful for ranking, e.g., in the case that a complete preorder is required. However, for choosing one 'best' alternative from a set of solutions, there is no need to specify this kind of preference information. Thus taking into account that the DM only has to provide preference information in form of pairwise comparisons, and without the need to infer inter-criteria information, we will apply the UTA$^{\text{GMS}}$ method as preference model in order to aid the DM with choosing a single, most preferred alternative.

Chapter 4

Case Study

The case study we describe in this paper is the case study as it is provided by Mallidis et al.. The single-objective optimization model is described in [46] and [45] contains the data for this case. There are some differences between the final paper [46] and the first report [45], in this research we follow the final paper. In this chapter we first describe the case study and the MOO model, then how it is implemented and finally we discuss the results of the multi-objective optimization and the preference model.

4.1 Description

The model considers a multinational company that aims to serve a specific geographical area (market) in the South East Europe region, trading various products with similar characteristics (e.g. white goods, furniture, etc.). For this supply chain, all cargo is transported from one major loading point far away to one of the entry points. These entry points are either international ports or other major transportation nodes and therefore have no capacity limitations. Through these entry points, the goods are transported to a distribution center for container deconsolidation purposes and from there the regional markets are served. The demand is allocated in these markets, so that they are the last stage of the supply chain. A simple (graphical) representation is shown in Figure 7.

In the above mentioned supply chain network, a number of decisions have to be made involving: (a) network design such as (i) the selection of entry points, (ii) the choice of transport means, (iii) the selection of distribution centers, and (iv) the determination of the associated flows; and (b) either leasing or outsourcing transportation modes.

The model has the following optimization criteria: (i) the total costs including (a) transportation/handling costs per TEU (Twenty feet Equivalent Unit; the size of a container), (b) operational costs of distribution centers, and (c) holding costs per TEU; and (ii) the total amount of emissions generated from the above supply chain operations separately for each type of emission.

We consider two realistic options for the supply chain designs: in the first option is the transportation outsourced to a third party logistics provider while in the second option leases the transportation. The distribution centers are leased through medium time-horizon contracts for both options. We make the following assumptions to model the supply chain: (i) rail services

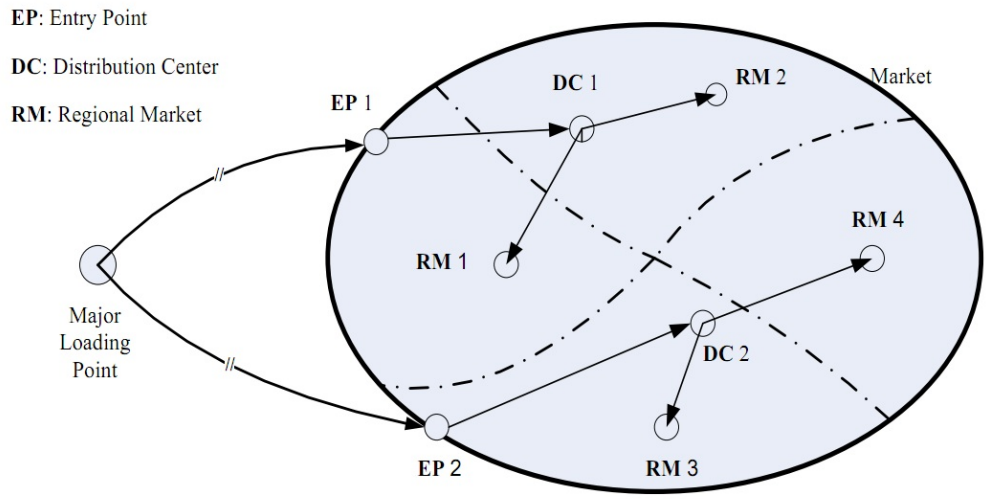


Figure 7: Supply Chain Network [46].

utilize the public railway network. This implies that rail transportation services are provided by each country's national railway company, so the option of leasing rail transportation is not considered valid. A block train is utilized when the number of TEUs exceed a specific number, which results in a discount cost per TEU transported; (ii) the outsourced transportation as well as the storage together with the deconsolidation/consolidation costs are charged per TEU based on spot market prices; and (iii) the trucks of a third party logistics provider (outsourcing transportation option) will transport cargo flows of other customers in the return haul of the trip, while leasing option trucks are exclusively utilized and thus will return empty or almost empty (e.g. carrying commercial returns, and/or packaging material). Therefore, the emissions of the outsourcing or leasing options in the return haul are treated in a different way.

The supply chain considered in this model for transporting white goods in the South Eastern Europe market that includes Bulgaria, Romania and Macedonia, has a planning horizon of one year, and the replenishment orders are set on a monthly basis. A market share of 20% is assumed of the annual sales of white goods. The major loading point (origin of the white goods) is the Port of Shanghai and we consider three different entry points: Ports of Thessaloníki, Varna, and Constanta. There are 16 potential distribution centers located on entry points (Thessaloniki, Varna and Constanta) and regional market's capital. There are 15 regional markets considered, satisfying the demand of the entire region. Figure 8 shows the supply chain for this region, where the triangles represent distribution centers or regional markets while circles represent the entry points.

The goods are first shipped from Shanghai to Gioia Tauro (Italy) with typical mother vessels (6000 TEU), where the containers are transshipped onto feeder vessels that deliver them at the entry points. The annual demand for the region of Bulgaria, Romania and Macedonia is based on (i) the estimated annual demand for the same products in Greece, and (ii) the ratio of each country's region GDP related to that of Greece.

To transport the goods from the entry points to the distribution centers, they use either: (i)

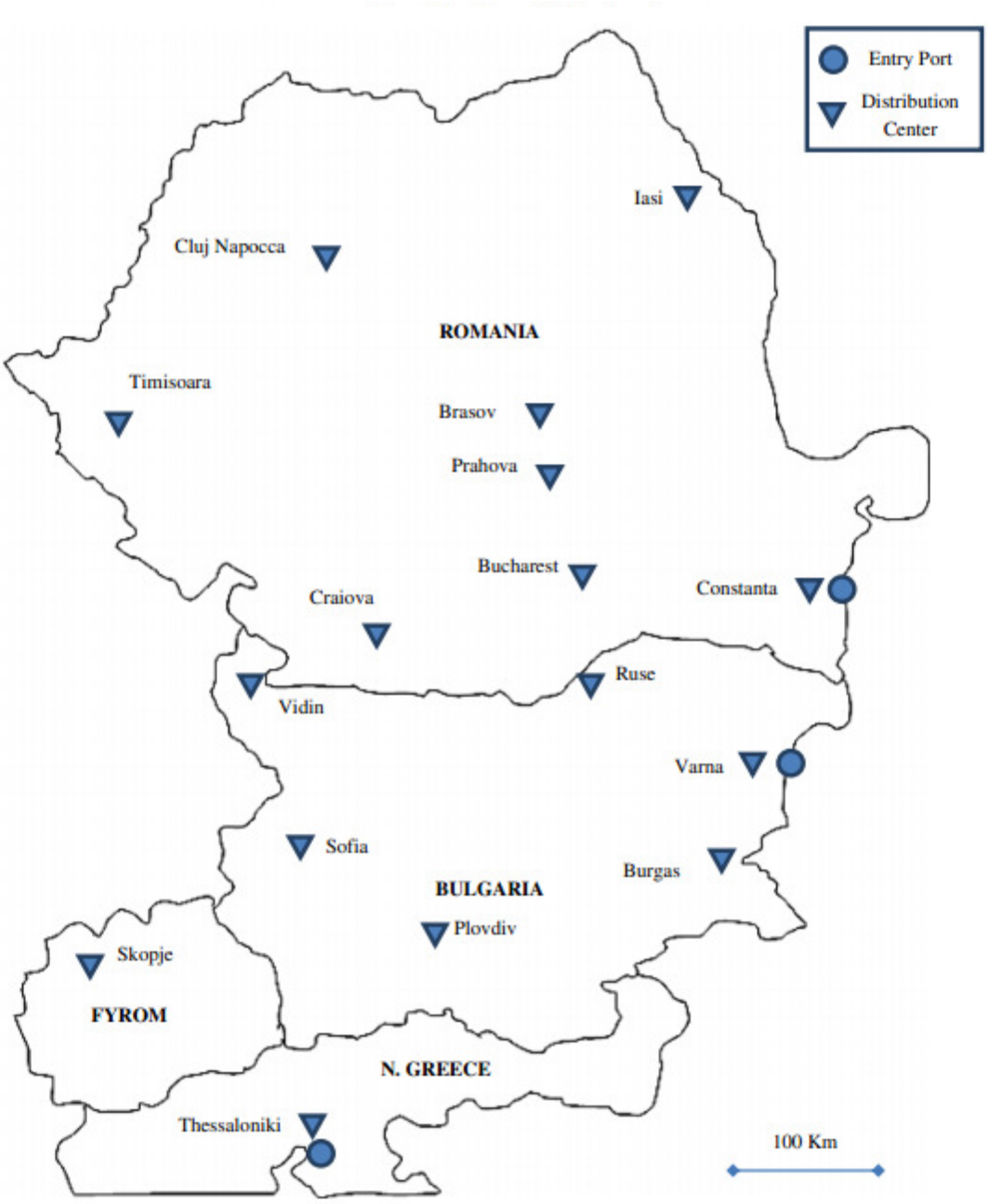


Figure 8: Supply Chain with possible Distribution Centers (triangles) and Entry Points (circles) [46].

electrical trains in the Bulgarian and Romanian rail routes, as well as the route from Thessaloniki to Skopje; (ii) diesel trains in the route of Thessaloniki to Kulatu/Promachon (national border); or (iii) heavy duty trucks in all routes (using truck types Euro III, IV, V, VI). To transport the goods from the distribution centers to the regional markets, delivery trucks are used since the transport to the regional market retail stores are Less than Full Truck Load (LTL). The difference in transportation times are insignificant since the major part of the total lead of maritime transportation is almost the same for all network realizations. Therefore, the holding costs are not included in this model.

Transportation costs

The regression equations 4.1 (heavy duty truck) and 4.2 (delivery truck) are used to calculate the transportation cost y in euro per 40ft container (2 TEU) for distance x in kilometers. For the option that leases the transportation, the freight rates for Euro III heavy duty trucks and delivery trucks are estimated to decrease 20% compared to outsourcing. This is because: (i) they are charged based on a fixed long term contract and not on a spot market basis, and (ii) no agency fees are included since the examined company has direct communication with the truck owners. The freight rates for the truck types Euro IV, V and VI are based on the Euro III trucks, the data can be found in [46].

$$y = 1.92x + 241.87 \quad 100 \text{ Km} \leq x \leq 1100 \text{ Km} \tag{4.1}$$

$$y = 1.953x + 1948 \quad 50 \text{ Km} \leq x \leq 600 \text{ Km} \tag{4.2}$$

The incorporated rail costs per 40ft container are: (i) a rail freight for transporting a 40ft container to its destinations rail depot, (ii) a rail freight rate per returning empty 40ft container, (iii) a fixed discharge from wagon and a loading on truck cost per 40ft container (€50) at the rail freight depots as also the loading of the returning empty 40ft container on the wagon, and (iv) a city limit expense of €100 for transporting a 40ft container by truck from the rail depot to its final destination and returning the empty container.

The costs of shipping per 40ft container have been retrieved from Orphee Beinoglou S.A. They include: (i) the fixed sea freight rates per 40ft container from Shanghai to the examined entry port; (ii) the local charges, e.g., discharge, pilotage costs and loading on a truck or wagon costs, at the entry ports; and (iii) custom clearance documentation costs at each port.

Transportation emissions

The emissions calculations are based on a fixed amount of CO_2 and PM emissions per ton/km obtained from [17]. The distance considered between two ports has been calculated using the Port to Port distance calculator[1]. Thus the choice of entry point only matters the feeder transport from Italy to the entry point.

[1] searates.com, `http://www.searates.com/reference/portdistance/`

The CO_2 and PM emissions for the Euro III heavy duty and delivery truck are calculated based on fixed amounts produced per ton/kilimoter ([8]). The same amount of CO_2 emissions will be considered for the Euro IV, V and VI heavy duty and delivery truck since a different Euro-type engines have no effect in terms of CO_2 emissions. Regarding PM emissions, significant reductions are observed between Euro III and IV heavy duty and delivery truck, and the Euro V and Euro VI, while there is no difference between the Euro IV and Euro V. In the option that involves outsourcing transportation is the company not accountable for the PM and CO_2 emissions generated in the return trips of the trucks, since they are typical utilized to transport cargo for other companies returning. Therefore, the amount of emissions generated in the return haul of the trip is not considered. For the leased trucks though, the emissions of the return trips are added because the trucks are dedicated to serve the company.

Rail CO_2 and PM emissions are calculated based on a fixed amount of CO_2 and PM emissions produced per ton/kilometer transported and incorporate the amount of emissions generated from the entry point to the distributions centers. In case of electrical trains, it depends on how the electricity is generated. As it is difficult to obtain good values for the region, the authors used published data from [17].

For both costs and emissions calculations and overviews of different routes, [46] and [45] contain more detailed information.

4.2 Model

The model we describe here is a Mixed Integer Linear Programming Problem as presented by [46]. It investigates potential entry points (from set EP), locations of the distribution centers (from set DC) and the transportation modes between the entry points and distribution centers, as well as from the distribution centers to the regional markets. In Mallidis et al.'s [46] model there are two kinds of objective functions: one for calculation the logistics costs and one for calculating different types of emissions, both to be minimized. The total logistics costs include: (i) the outsourced or leased transportation and the associated handling costs per TEU which vary for the different transportation modes, (ii) the custom formalities expenses per TEU, (iii) the holding costs for the cycle stocks per TEU, and (iv) the costs for outsourcing storage and deconsolidation/consolidation of services to a third party logistics company. The emissions data are available for CO_2, CO, PM, SO_2 and NO_x (from the set EG), but only CO_2 and PM (Particulate Matter) are taken into account. Table 1 provide the variables for the model and Table 2 the parameters.

Table 1: Decision variables

Variable	Description
x_{ij}^{m}	number of TEU transported from node i to node j using transportation mode m =1,..,M.
z_{ij}	binary variable which indicates whether a block train is utilized or not in the route from node i to node j.
y_{j}^{w}	binary variable which indicates whether a distribution center of size w is leased at node j or not.

Table 2: Model parameters

Parameter	Description
D_r	total demand at regional market r.
c_{ij}^{m}	cost of transporting a TEU from node i to node j using transportation mode m (node 0 is the major loading port).
c_{ij}^{bt}	block train transportation cost from node i to node j per TEU.
c_{j}^{dc}	deconsolidation/consolidation cost per TEU at a distribution center at node j (only in the option of outsourcing).
${}^{g}e_{ij}^{m}$	emissions of type g generated from transporting a TEU from node i to node j using transportation mode.
${}^{g}e_{ij}^{bt}$	emissions of type g generated during a block train trip from node i to node j.
L^{w}	capacity of a distribution center of size w (L^{w} is considered infinite).
t_{ij}^{m}	transportation time from node i to node j using transportation mode m.
h	holding cost per TEU.
N	represents the minimum TEU volume for charging a block train.
M_0	represents a very large constant.

Consequently Mallidis et al. (2010a) suggested the following integer programming model:

Minimize total cost (TC):

$$\begin{aligned} TC = & \sum_{i\in EP}\sum_{m=1}^{M} c_{0i}^{m}x_{0i}^{m} + \sum_{i\in EP} c_{0i}^{bt}x_{0i}^{bt} + h\sum_{i\in EP}\left(t_{0i}^{bt}x_{0i}^{bt} + \sum_{m=1}^{M} t_{0i}^{bt}x_{0i}^{bt}\right) \\ & + \sum_{i\in EP}\sum_{j\in DC}\sum_{m=1}^{M}\left(c_{ij}^{m} + c_{j}^{dc}\right)x_{ij}^{m} + \sum_{i\in EP}\sum_{j\in DC}\left(c_{ij}^{bt} + c_{j}^{dc}\right)x_{ij}^{bt} + \\ & \sum_{i\in EP}\sum_{j\in DC}\left(t_{ij}^{bt}x_{ij}^{bt} + \sum_{m=1}^{M} t_{ij}^{m}x_{ij}^{m}\right) + \sum_{w}\sum_{j\in DC} f_{j}^{w}y_{j}^{w} + \sum_{j\in DC}\sum_{r\in RM}\sum_{m=1}^{M} c_{jr}^{m}x_{jr}^{m} \\ & + \sum_{j\in DC}\sum_{r\in RM}\left(t_{jr}^{bt}x_{jr}^{bt} + \sum_{m=1}^{M} t_{jr}^{m}x_{jr}^{m}\right) \end{aligned} \tag{4.3}$$

Or Minimize total emissions (TE_g) of type g:

$$TE_g = \sum_{i\in EP}\sum_{m=1}^{M} {}^{g}e_{0i}^{m}x_{0i}^{m} + \sum_{i\in EP} {}^{g}e_{0i}^{bt}x_{0i}^{bt} + \sum_{i\in EP}\sum_{j\in DC}\sum_{m=1}^{M} {}^{g}e_{ij}^{m}x_{ij}^{m} + \sum_{i\in EP}\sum_{j\in DC} {}^{g}e_{ij}^{bt}x_{ij}^{bt} + \sum_{j\in DC}\sum_{r\in RM}\sum_{m=1}^{M} {}^{g}e_{jr}^{m}x_{jr}^{m} + \sum_{j\in DC}\sum_{r\in DC} {}^{g}e_{jr}^{bt}x_{jr}^{bt} \qquad \forall g \in \mathbf{EG}. \tag{4.4}$$

Subject to:

Flow Constraints

$$\sum_{m=1}^{m} x_{0i}^{m} + x_{0i}^{bt} = \sum_{j\in DC}\sum_{m=1}^{M} x_{ij}^{m} + \sum_{j\in DC} x_{ij}^{bt}, \qquad \forall i \in EP \tag{4.5}$$

$$\sum_{i\in EP}\sum_{m=1}^{M} x_{ij}^{m} + \sum_{i\in EP} x_{ij}^{bt} = \sum_{r\in RM}\sum_{m=1}^{M} x_{jr}^{m} + \sum_{r\in RM} x_{jr}^{bt}, \qquad \forall j \in DC \tag{4.6}$$

$$\sum_{j\in DC}\sum_{m=1}^{M} x_{jr}^{m} + \sum_{j\in DC} x_{jr}^{bt} = D_r, \qquad \forall r \in RM \tag{4.7}$$

Capacity Constraints

$$\sum_{i\in EP}\sum_{m=1}^{M} x_{ij}^{m} + \sum_{i\in EP} x_{ij}^{bt} \leq \sum_{w} L^{W} y_{j}^{w}, \qquad \forall j \in DC \tag{4.8}$$

$$\sum_{w} y_{j}^{w} \leq 1, \qquad \forall j \in DC \tag{4.9}$$

Block Train Constraints

$$x_{0i}^{bt\smile} M_0 z_{0i} \leq 0, \qquad \forall i \in EP \tag{4.10}$$

$$x_{0i}^{bt\smile} N z_{0i} \geq 0, \qquad \forall i \in EP \tag{4.11}$$

$$x_{ij}^{bt\smile} M_0 z_{ij} \leq 0, \qquad \forall i \in EP, \forall j \in DC \tag{4.12}$$

$$x_{ij}^{bt\smile} N z_{ij} \geq 0, \qquad \forall i \in EP, \forall j \in DC \tag{4.13}$$

$$x_{jr}^{bt\smile} M_0 z_{jr} \leq 0, \qquad \forall j \in DC, \forall r \in RM \tag{4.14}$$

$$x_{jr}^{bt\smile} N z_{jr} \geq 0, \qquad \forall j \in DC, \forall r \in RM \tag{4.15}$$

Non-Negativity Constraints

$$x_{ij}^{m} \geq 0 \tag{4.16}$$

The flow constraints guarantee that there is no product surplus or shortage in the supply chain, in other words you can not deliver more goods than that have entered in the entry point, distribution center or regional market. More specifically, Eq. 4.5, 4.6 and 4.7 guarantee the balance between the inbound and outbound of the entry point and distribution center, the distribution center and the regional market, and it makes sure that the amounts that entered the regional market are equal to the demand of that region. Capacity constraint 4.8 guarantees that the capacity will be adequate to handle the flow that will pass through it if the distribution center is activated and constraint 4.9 makes sure there is only one distribution center per possible location. The block train constraints guarantee that a block train is used when the amount of TEUs are above a specific number. The model is an extension of a two-level capacitated location problem, extended with an extra objective and different transportation modes with block train requirements.

4.3 Implementation

In this section we describe briefly how different parts are implemented before we elaborate our results. The scalarizing methods are compared with genetic algorithms, and both try to approximate the Pareto front. UTA$^{\text{GMS}}$ is used to find solutions which best fit to some preference statements.

4.3.1 Scalarizing methods

The scalarizing methods, i.e. the Weighted Sum Method (WSM) and the ε-Constraint Method (ECM), are implemented using Lindo Lingo software[2]. In order to obtain a front for the WSM, we used all weight combination from the range of 0.01 to 1 with steps of 0.01. For the ECM we took nine ε values between the objective's best value to its worst from each objective.

[2]more information available at `http://www.lindo.com/`

4.3.2 Genetic Algorithm

The genetic algorithms have been implemented using the ParadisEO framework[3]. This framework allows you to use different MOEA such as NSGA, NSGA-II, IBEA and SPEA2, and just run the procedures with some parameters and implementation of the initialization, evaluation and genetic operators. With this framework, one can easily switch between genetic algorithms rather than implement the algorithms themselves. The source code of the ParadisEO implementation is available at github: `https://github.com/CornePlas/MOOSCN`.

The ParadisEO framework is required to run this optimization problem. ParadisEO 1.3-beta2 is used, see `http://paradiseo.gforge.inria.fr/` for more information and downloads.

Representation

To involve multiple stages, we use the reversed supply chain, i.e., we start with the last stage and end with the first stage. Thus in our case, first the demand of the regional markets (depots) will be allocated over the distribution centers (sources) using their priorities to set up the transportation tree. Then we can continue with the distribution centers as depots using their output to the regional markets as demand, and the entry points as sources. There is only one possibility to supply these entry points, thus from the major loading point to the entry points it is not necessary to represent the transport flows in the chromosomes. This means that the final chromosome consists of four parts:

1. Integers (0 – 4/5) of size $|EP|+|DC|$ representing transportation modes for the first stage;
2. Permutation of size $|EP| + |DC|$ with priorities to set-up the transportation tree for the first stage;
3. Integers (0 – 4) of size $|DC| + |RM|$ representing the transportation modes for the second stage;
4. Permutation of size $|DC| + |RM|$ with priorities to set-up the transportation tree for the second stage.

Thus if we translate this to our case; the chromosome length, consisting of four parts, is equal to $(|EP| + |DC|) + (|EP| + |DC|) + (|DC| + |RM|) + (|DC| + |RM|) = (3 + 16) + (3 + 16) + (16 + 15) + (16 + 15) = 100$. This means that we can represent each possible solution by a chromosome with the length of 100 integers and no need for binary encoding.

Algorithm: genetic operators

On the parts that contain the transportation modes, default crossover and mutation operators can be applied. Priority-based encoding belongs to the permutation encoding class and thus require specific operators, e.g., for crossover partially mapping crossover, order crossover or position based crossover.

We used different crossover operators for the different parts: the transportation parts used uniform crossover, i.e. that each digit has a certain chance, in our case 50%, to be exchanged. For

[3]available at `http://paradiseo.gforge.inria.fr/`

the priorities parts we used order crossover which basically changes order of the chromosomes, but maintain unique priorities.

Mutation does not involve a second chromosome, but changes the position(s) of digits. We used the same mutation operators for the different parts. These are swap mutation, shift mutation and TwoOpt mutation. The swap mutation swaps two components of a chromosome, shift mutation shifts two components of a chromosome, and TwoOpt mutation takes two random points and between these points it takes the inverse order.

4.3.3 UTA$^{\mathbf{GMS}}$

To test whether UTA$^{\mathrm{GMS}}$ is applicable to this case we have designed a test which incorporates:

- Different (sample) sizes/number of alternatives $\{20, 40, \ldots, 180, 200\}$;
- Different number of statements per iteration (5, 10, 20);
- Measures how much computation time it takes to remain with one alternative left (total time);
- Time needed per iteration;
- Count the number of statements used and the number of iterations required.

For the smaller sets (up to $N = 60$), we iterate until we end up with one alternative left using different sizes of preference statements. For each run, either five, ten or twenty preference statements (P) are used per iteration (only if there are enough alternatives left). Then after each iteration a number of alternatives can be removed that are dominated according the calculations using UTA$^{\mathrm{GMS}}$. For the remaining alternatives P random preference statements are added. Then for each case, it is measured how long it takes to do the calculations and end up with one remaining alternative, how much time the calculations for the necessary relations takes, how many preference statements are used and how many iterations are needed. For population sizes larger than 60 we only measure how much time it takes for the first run and then we can approximate the total time required to finish with one solution. An overview of the test procedure is presented in Algorithm 4. The full R-script is in Appendix 6.2 and also available at github: `https://github.com/CornePlas/MOOSCN-R`.

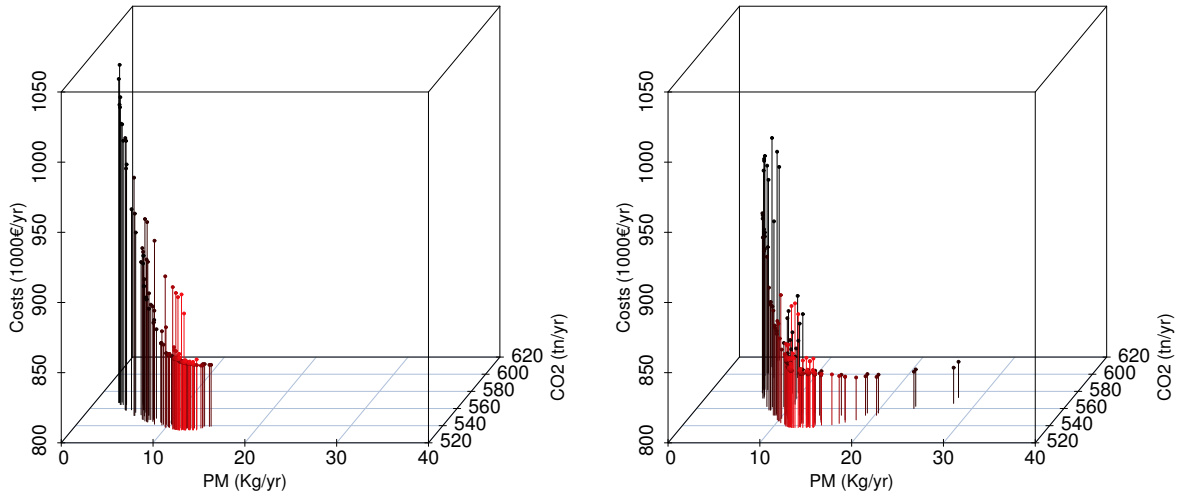
The tests are executed 16 times with R (version 2.10.1 (2009-12-14)), using eight Intel(R) Xeon(R) CPU X3440 @ 2.53GHz processors. The R-script is not multi-threaded, so each run uses a single core and is run in parallel.

4.4 Results

In this section we elaborate the results. First we show the Pareto front using scalarizing methods and genetic algorithm, and then the results of UTA$^{\mathrm{GMS}}$ applied on solutions from both methods. Option A represents the solutions with outsourcing the transportation services, while Option B represents the leasing of the transportation services.

Algorithm 4 Loop to test UTA^{GMS}

$N \leftarrow \{10, 20, 30, 40, 50, 60\}$
$P \leftarrow \{5, 10, 20\}$
for all $i \in N$ **do**
  for all $j \in P$ **do**
    while $|N_i| > 1$ **do**
      If $P_j > N_j, P_j = P_j/2$;
      Add P_j random preference statements, only if $P_j < N_j$, else $P_j = P_j/2$;
      Calculate necessary relation;
      Remove solutions from N_i that are definitely not the preferred one.
    end while
  end for
end for



(a) Option A

(b) Option B

Figure 9: Pareto front of the weighted sum method.

4.4.1 Scalarizing methods

Weighted Sum Method

Here we will discuss the results we have obtained from apply the Weighted Sum Method to the case. Figure 9a (Option A) shows the results for outsourcing transportation, while Figure 9b (Option B) shows the results for leasing transportation. As we can see from these figures, there is a pay-off between the three different optimization criteria. If you want to decrease the PM emissions, we see an increase of the costs and the other way around. The same fact applies to the CO_2 emissions and the PM emissions; decreasing one results to an increase of the other one.

We can translate these results into tables, where we can quantify the differences related to one objective. We have taken the single-objective optimization of the costs as bases to see the increase/decrease of the objectives. Table 3 shows this for Option A, and Table 4 for Option B. In these tables are the costs measured in €1000 per year, CO_2 emissions are measured in tons per year and PM emissions are measured in Kg per year. The last two columns show some

Table 3: Trade-offs for option A (WSM)

ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
1	843.70		539.43		14.81		3	12
2	843.70	0.00%	539.43	0.00%	14.81	0.00%	3	14
32	845.97	0.27%	537.17	−0.42%	12.41	−16.23%	3	14
82	848.27	0.54%	535.61	−0.71%	12.02	−18.82%	3	14
132	851.35	0.91%	538.88	−0.10%	10.22	−30.97%	3	14
167	881.94	4.53%	547.01	1.40%	7.65	−48.36%	3	14
183	904.35	7.19%	567.43	5.19%	5.26	−64.47%	3	14
195	942.95	11.76%	558.49	3.53%	4.66	−68.54%	3	14
205	997.98	18.29%	567.43	5.19%	2.94	−80.13%	3	14
212	1039.87	23.25%	568.18	5.33%	2.61	−82.39%	3	14

Table 4: Trade-offs for option B (WSM)

ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
1	825.47		572.80		27.53		3	11
2	825.47	0.00%	565.96	−1.20%	27.53	0.00%	3	13
44	830.90	0.66%	552.11	−3.61%	11.97	−56.54%	3	13
87	834.76	1.13%	543.97	−5.03%	12.59	−54.28%	3	13
111	836.61	1.35%	548.41	−4.26%	10.79	−60.80%	3	14
165	841.75	1.97%	539.54	−5.81%	11.66	−57.64%	3	13
219	891.03	7.94%	543.36	−5.14%	10.46	−61.99%	3	13
220	901.69	9.23%	581.82	1.57%	6.06	−77.99%	3	13
244	956.64	15.89%	598.22	4.44%	4.47	−83.78%	3	13
245	963.16	16.68%	608.52	6.23%	4.44	−83.88%	3	13

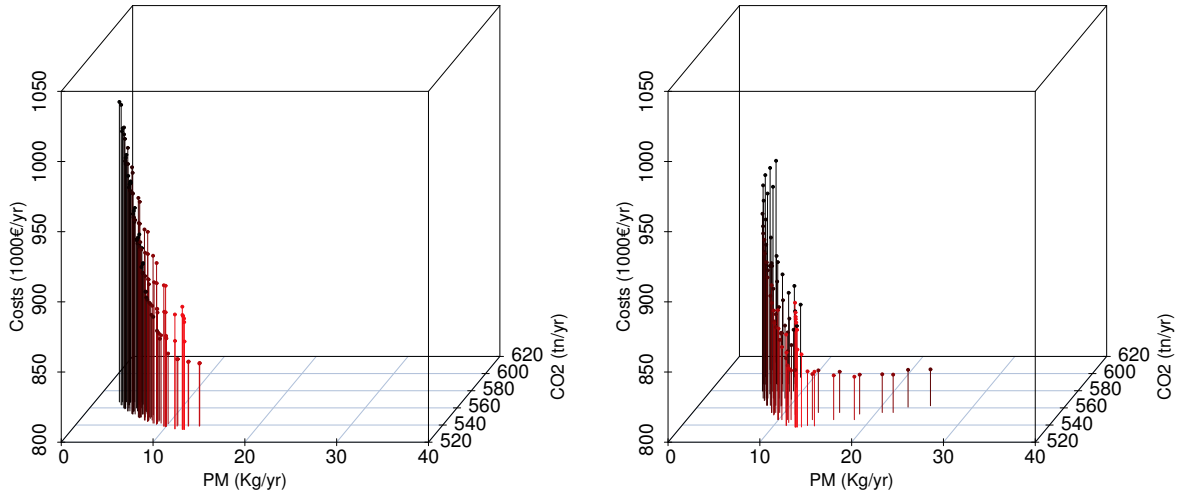
differences in the design of the network by showing the number of entry points and distribution centers that are used to fulfill the demand. Note that the tables contain only a part of obtained front.

To obtain these fronts, 5151 optimization problems need to be solved, i.e. each weight combination of three in $A = \{1, 0.99, \ldots, 0.01, 0\}$, thus we optimize the problem with weight combinations such as $\{1, 0, 0\}, \{0.99, 0.01, 0\}, \{0.99, 0, 0.01\}$, etc., this implies that we need $1 + 2 + 3 + 4 + \ldots + n = (n^2 + n)/2$ problems to optimize, $n = |A| = 101 \implies (101^2 + 101)/2 = 5151$ required optimizations. Each optimization takes about 2-3 seconds, which means that the total computation time for obtaining a front with the weighted sum method is about four hours.

ε-Constraint Method

Here we show the same figures and tables for the ε-Constraint Method (ECM) as we did for the Weighted Sum Method (WSM). Figures 10a and 10b show the graphs, and Table 5 and 6 are the tables. Similar to the WSM, these tables show only a few points of the graphs. More solutions for the ECM can be obtained by adding runs with different ε-values. However, it is impossible to know the right ε-values beforehand, so it is hard to find a good representation of the whole Pareto front.

Compared to the weighted sum method, the ECM took less time to obtain a Pareto front.



(a) Option A

(b) Option B

Figure 10: Pareto fronts for the ε-constraint method.

Table 5: Trade-offs for option A (ε-CM)

ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
1	844.702		539.17		13.59		3	12
2	844.702	0.00%	539.17	0.00%	13.59	0.00%	3	14
15	849.961	0.62%	541.62	0.45%	9.96	−26.71%	3	15
29	862.6	2.12%	538.65	−0.10%	10.00	−26.44%	3	11
41	876.426	3.76%	534.95	−0.78%	12.23	−10.01%	3	14
58	884.489	4.71%	557.21	3.35%	6.33	−53.42%	3	12
84	919.3	8.83%	560.54	3.96%	5.12	−62.33%	3	10
101	938.2	11.07%	555.60	3.05%	5.12	−62.33%	3	14
121	976	15.54%	559.95	3.85%	3.87	−71.55%	3	12
130	1013.8	20.02%	566.91	5.14%	2.68	−80.26%	3	11

Table 6: Trade-offs for option B (ε-CM)

ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
1	825.93		562.48		25.28		3	15
2	826.55	0.07%	560.95	−0.27%	22.96	−9.18%	3	14
16	832.51	0.80%	546.25	−2.89%	13.68	−45.89%	3	11
32	851.76	3.13%	537.76	−4.40%	13.15	−47.98%	3	12
46	864.89	4.72%	539.36	−4.11%	11.36	−55.06%	3	14
64	878.48	6.36%	537.76	−4.40%	12.51	−50.51%	3	14
73	891.15	7.90%	576.30	2.46%	6.72	−73.42%	3	13
84	917.41	11.08%	579.65	3.05%	5.71	−77.42%	3	13
90	943.67	14.26%	596.35	6.02%	4.65	−81.59%	3	12
92	943.67	14.26%	613.05	8.99%	4.53	−82.09%	3	13

The method took about 45 minutes for each option to obtain a front. However, this method had fewer optimization problems to solve ($9^3 = 729$) and many of the (single) ε-constraint problems did not have a feasible solution. Also, the ECM obtained less solutions compared to the WSM. The ECM finds 130 solutions for option A and 92 for option B, while the WSM finds 212 solutions for option A and 245 for option B. To obtain a larger representation, more computation time is needed with additional ε-values.

4.4.2 Genetic Algorithm

Both genetic algorithms have been run with a population size of 400, for 10^4 generations with a probability of 70% for crossover and a probability of 30% for mutation. It takes about one hour to obtain a Pareto front with these parameters for the NSGA-II, while it takes about six hours for SPEA2 (with an unbounded archive, the computation time is similar for an archive size equal to the initial population). Then the resulting fronts contain between 200 and 230 possible solutions for NSGA-II and between 1300 and 2000 solutions for SPEA2 (or equal to the archive size).

NSGA-II

As we already have seen Pareto fronts for the scalarizing methods, we focus here on the comparison of methods instead of indicating that decreasing PM will lead to an increase of costs.

Figure 11a and 11b shows one example of the Pareto front generated with the NSGA-II method. Depending on parameters such as the population size, number of generations, and crossover and mutation percentage, the Pareto front can differ. Even if you use the same parameters for each run, the front can differ due to used pseudo-randomness as well as for the initialization of the population and the genetic operators. Table 7 and 8 show some possible solutions obtained with the algorithm. Remarkable about these configurations is that the extreme points (obtained with the scalarizing methods) are not included, but close approximates of these points are obtained (for the region it converges to). Also, NSGA-II is converging to the first objective in this particular case. This means that it find a lot of solutions near the optimum for costs, but totally lack of efficient solutions with low PM values. Thus the obtained fronts with NSGA-II are not well-spread. However, it is possible to obtain well-spread solutions with NSGA-II, if you decrease the amount of generation to, e.g., 1000 generations. In that case though, the solutions are not really efficient compared to fronts obtained with the scalarizing methods.

SPEA2

The results we see with the SPEA2 algorithm are similar to the results obtained with the NSGA-II algorithm using a low amount of generations. Figures 11c and 11d show the Pareto front obtained with the SPEA2 algorithm and Tables 9 and 10 show some example points. The figures of NSGA-II and SPEA2 are very different; whereas SPEA2 maintain a well-spread distribution of the front, the NSGA-II is focused on the low costs/CO_2 only, with high PM. The

Table 7: Trade-offs for option A (NSGA-II)

ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
213	844.58		538.95		29.12		3	14
211	844.71	0.02%	538.69	−0.05%	28.95	−0.56%	3	15
199	845.18	0.07%	537.55	−0.26%	30.22	3.79%	3	14
145	846.21	0.19%	537.15	−0.33%	28.49	−2.15%	3	15
130	846.59	0.24%	537.15	−0.33%	28.08	−3.58%	3	15
109	847.27	0.32%	536.33	−0.49%	30.46	4.63%	3	13
101	847.43	0.34%	537.15	−0.33%	27.00	−7.27%	3	13
80	848.97	0.52%	536.14	−0.52%	26.45	−9.16%	3	14
2	861.76	2.03%	538.55	−0.07%	22.68	−22.11%	3	14
1	882.20	4.45%	545.02	1.13%	22.66	−22.17%	3	13

Table 8: Trade-offs for option B (NSGA-II)

ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
216	826.92		560.14		38.63		3	14
215	827.26	0.04%	560.14	0.00%	38.02	−1.57%	3	14
190	828.25	0.16%	553.62	−1.16%	33.93	−12.17%	3	14
173	829.68	0.33%	550.26	−1.76%	36.32	−5.99%	3	13
142	830.58	0.44%	548.20	−2.13%	35.31	−8.59%	3	15
112	832.56	0.68%	544.84	−2.73%	35.80	−7.33%	3	13
86	833.67	0.82%	543.56	−2.96%	35.32	−8.56%	3	14
44	835.71	1.06%	547.03	−2.34%	30.11	−22.06%	3	13
2	840.22	1.61%	541.31	−3.36%	30.75	−20.39%	3	14
1	842.84	1.92%	547.77	−2.21%	28.51	−26.19%	3	14

obtained solutions with SPEA2, are however, less efficient than the solutions obtained with the scalarizing methods.

The SPEA2 algorithm described by Zitzler et al. uses a bounded archive size. ParadisEO [44] provide, however, to use an unbounded archive size. With the unbounded archive size, the number of resulting solutions is increasing if the amount of generations increases. Then because the population size increase, the computation time is also increasing. Therefore, you can restrict the population size and resulting solutions by using a fixed archive size. This will limit the computation time also. The results presented here do not use a fixed archive size.

4.4.3 The resulting Pareto front

In the previous sections we described each option separately for each method to see the differences between both options. However, in that case should the DM choose one option explicitly beforehand. Therefore, we take the fronts obtained with the weighted sum method and combine both options to represent the DM a single front. Figure 12 shows the resulting front. In this figure are also the extreme points from Mallidis et al. [46] included, to see how the solutions relate to the extreme ones.

If you compare the solutions, you see on the left side of the figure solutions from option A, while on the right side you find most solutions from option B. This means that the transportation

Table 9: Trade-offs for option A (SPEA2)

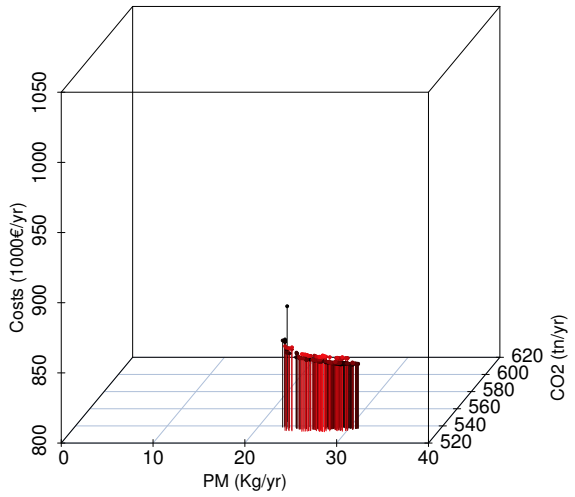
ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
1943	850.04		557.30		26.65		3	14
1900	856.35	0.74%	534.82	−4.03%	26.78	0.50%	3	12
1645	883.24	3.91%	539.49	−3.20%	12.88	−51.67%	3	12
1574	894.28	5.21%	572.02	2.64%	6.30	−76.34%	3	14
1442	912.95	7.40%	550.02	−1.31%	4.42	−83.40%	3	13
784	870.13	2.36%	541.41	−2.85%	15.35	−42.41%	3	14
365	927.49	9.11%	552.94	−0.78%	3.93	−85.27%	3	13
32	993.04	16.82%	551.93	−0.96%	2.55	−90.42%	3	12
25	994.83	17.03%	555.45	−0.33%	2.43	−90.90%	3	14
2	1011.98	19.05%	552.46	−0.87%	2.35	−91.19%	3	13

Table 10: Trade-offs for option B (SPEA2)

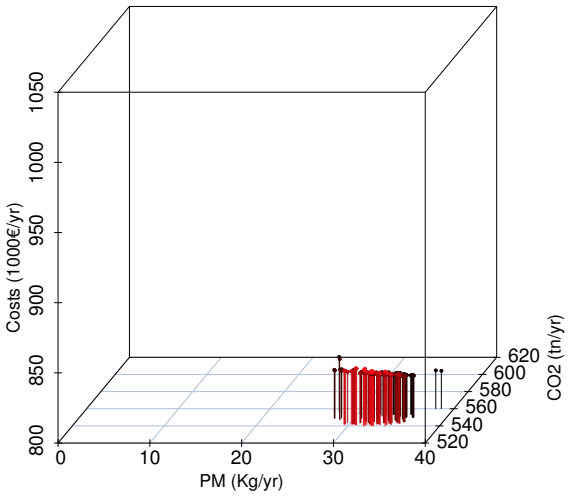
ID	Costs	ΔC	CO_2	ΔCO_2	PM	ΔPM	EPs	DCs
1384	835.52		574.15		39.84		3	14
1376	842.65	0.85%	549.58	−4.28%	24.88	−37.56%	3	15
1346	849.08	1.62%	541.61	−5.67%	29.43	−26.12%	3	12
778	885.97	6.04%	575.51	0.24%	11.15	−72.00%	3	13
745	888.91	6.39%	554.48	−3.43%	17.74	−55.48%	3	15
591	900.08	7.73%	560.60	−2.36%	15.17	−61.93%	3	15
590	900.18	7.74%	584.76	1.85%	9.52	−76.11%	3	12
134	936.95	12.14%	596.73	3.93%	4.89	−87.72%	3	15
9	947.44	13.40%	600.96	4.67%	4.35	−89.09%	3	12
1	950.49	13.76%	599.28	4.38%	4.44	−88.84%	3	13

is outsourced for solutions that try to minimize the PM emissions, while the transportation is leased for solutions that minimizes costs (and CO_2). The front presented in Figure 12 contains all solutions obtained with the generation of the Pareto front for option B, while solutions from option A are either beyond the PM emission's lower bound of option B or it adds a solution with a lower amount of CO_2 emission for a certain level of PM emissions.

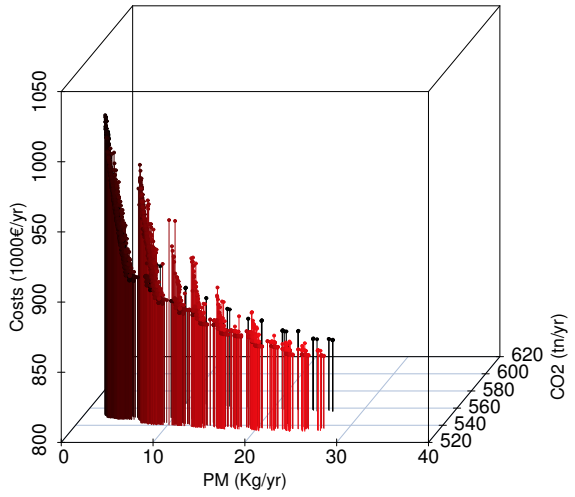
Figure 13 represents the same Pareto front in a two dimensional surface, where the colors indicate the third dimension and the circles represent the solutions. The costs and CO_2 are on the axes, while the legend shows the colors that belong to the different levels of PM emissions. This figure may give a better view on how costs, CO_2 and PM relate to each other.



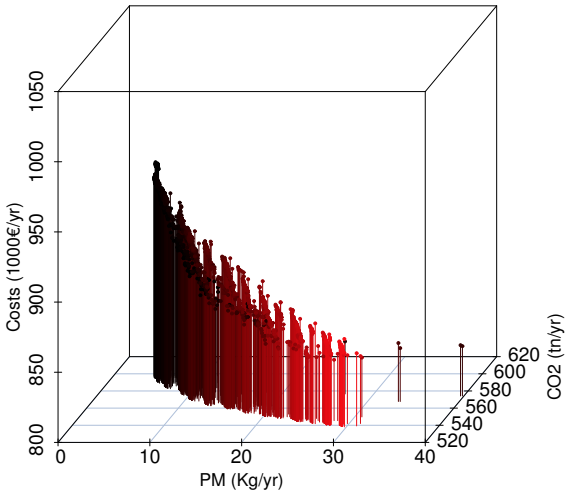
(a) Option A - NSGA-II



(b) Option B - NSGA-II



(c) Option A - SPEA2



(d) Option B - SPEA2

Figure 11: Pareto fronts for GAs.

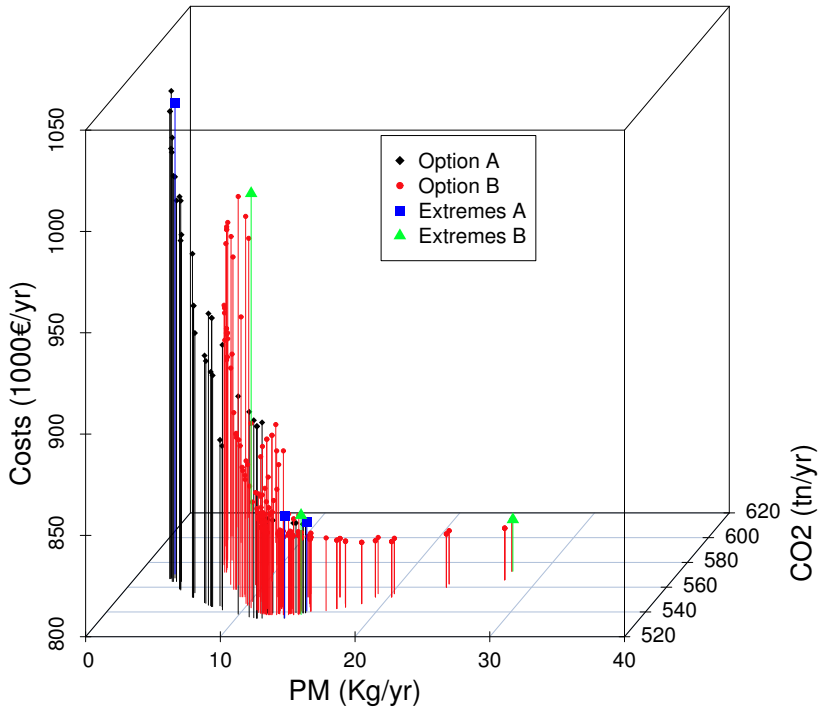


Figure 12: The resulting Pareto front

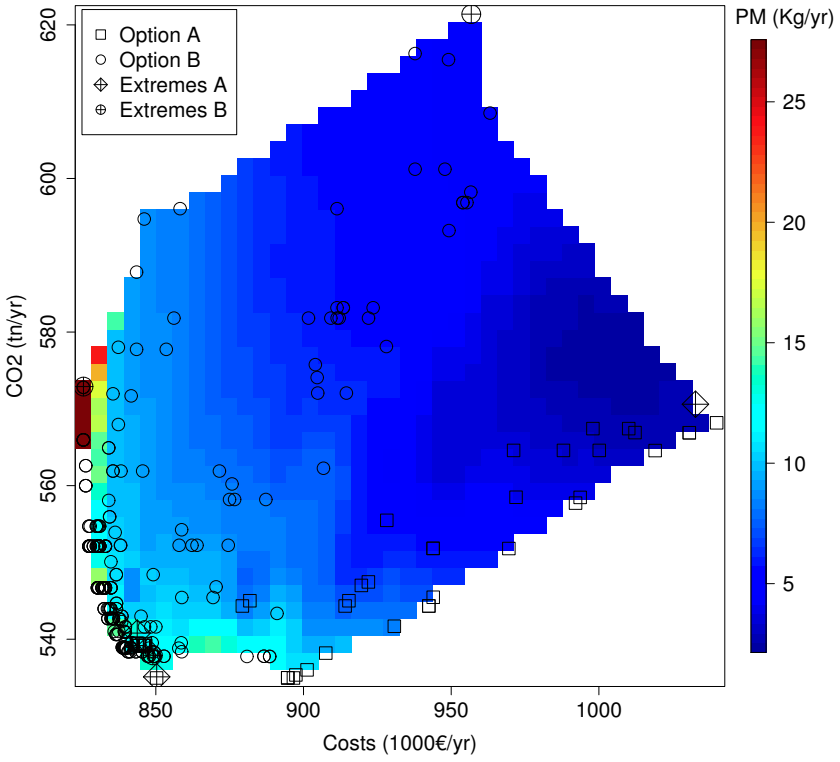
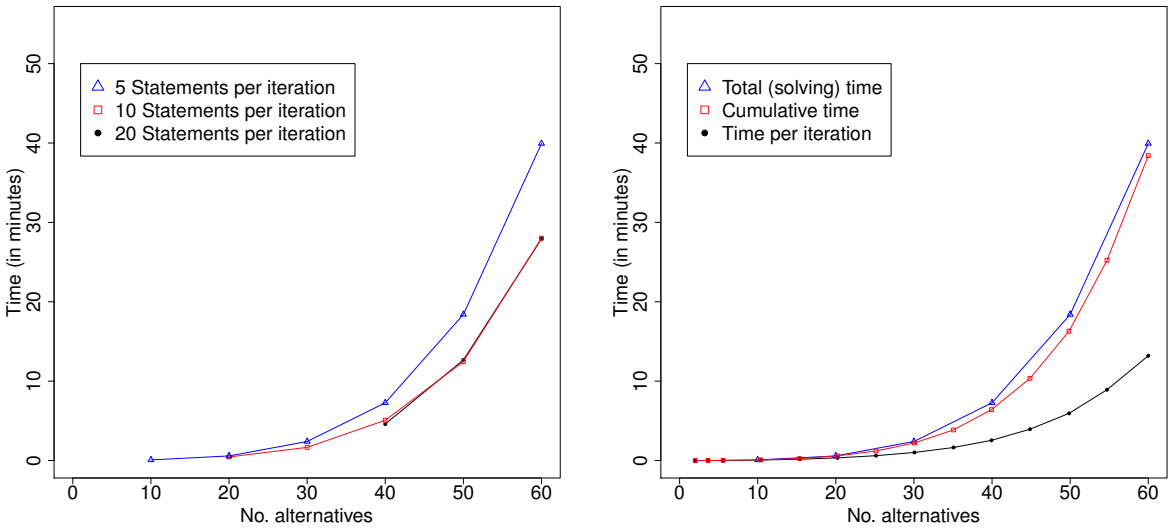


Figure 13: 2-d heat graph of Pareto front.



(a) Total computation time

(b) Total time vs iteration time

Figure 14: UTA$^{\text{GMS}}$ Results.

4.4.4 UTA$^{\text{GMS}}$

We tested the method with different sizes, i.e. number of alternatives available as possible solution/configuration. Also, we have applied different fixed size of preference statements required per iteration to show the differences between them in computation time. Figure 14a shows the results with sample sizes up to 60. In Figure 14b we added the iterative process in which 60 alternatives are reduced to a single solution. It shows that the cumulative time per iteration is similar to previous total times of smaller sample sizes. Therefore, we can approximate the total time required for larger sample sizes with just calculating the first iteration, i.e. calculate for example for 200 alternatives only the first run with 5, 10, or 20 preference statements.

If there are n alternatives, n^2 calculations are required to obtain a matrix in which relations for both $(a \succ^N b)$ and $(b \succ^N a)$ are calculated. The more preference statements are added, the longer it takes to do all computations. However, the total run time decreases if there are more preference statements added per iteration because you can eliminate more solutions after each iteration and thus fewer iterations are needed.

In Figure 16 the approximations are made for the total calculation time for amount of alternatives up to 200, based on the single run time in Figure 15 and assuming that for each iteration the number of preference statements determines how many alternatives can be removed after each iteration. Thus if we start with 200 alternatives and use 20 preference statements per iteration, we need the cumulative sum of the single run times of the set $\{200, 180, 160, \ldots, 40, 20, 10, 5, 3, 2\}$. These cumulative sums show also what (total) time is needed for the intermediate numbers of alternatives.

Tables 11 and 12 show the data presented in the figures. The mean is the average time UTA$^{\text{GMS}}$ took to do the calculations, and the last column shows the standard deviation of the 16 runs.

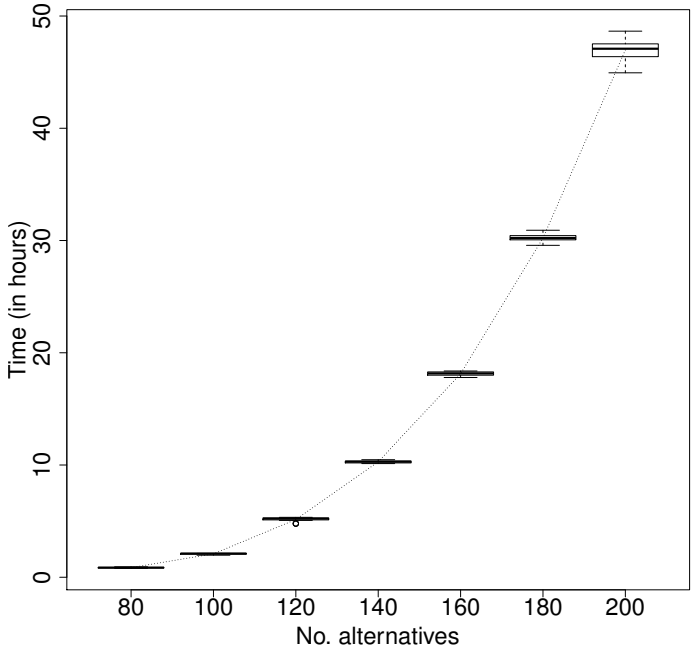


Figure 15: UTA$^{\mathrm{GMS}}$ results for large sample sizes.

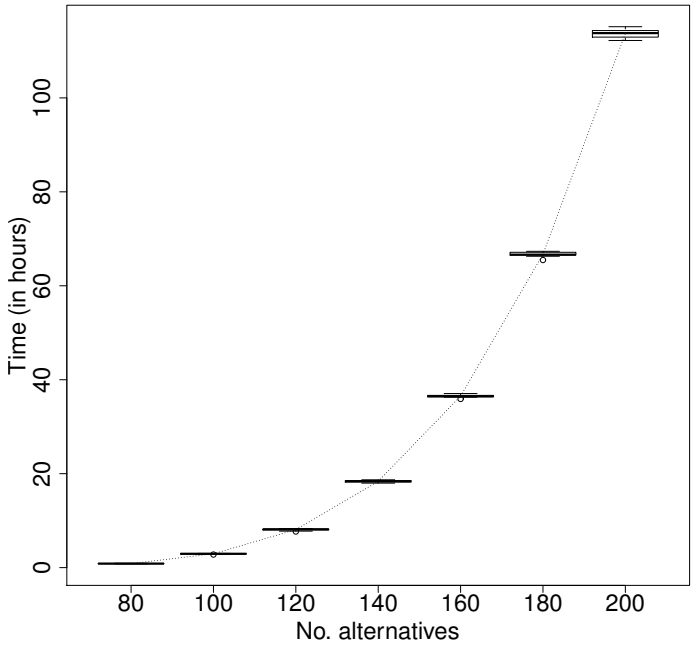


Figure 16: UTA$^{\mathrm{GMS}}$ results with approximations for large sample sizes.

Table 11: UTA results for small samples (Mean and Std. Dev. in seconds)

Alternatives	**Pref. Statements**	**Mean**	**Standard Deviation**
10	5	4.666	0.182
20	5	34.738	1.747
20	10	27.019	2.453
30	5	143.950	6.195
30	10	99.734	4.057
40	5	436.396	22.068
40	10	304.482	7.619
40	20	275.440	19.977
50	5	1103.902	57.020
50	10	747.597	28.370
50	20	760.285	56.424
60	5	2396.269	126.376
60	10	1680.253	82.889
60	20	1675.409	95.921

Table 12: UTA results for large samples (Mean and Std. Dev. in seconds)

Alternatives	**Pref. Statements**	**Mean**	**Standard Deviation**
80	20	3086.998	125.840
100	20	7510.208	161.993
120	20	18524.308	693.801
140	20	37030.249	392.654
160	20	65273.258	659.275
180	20	108787.044	1372.603
200	20	169123.369	3420.303

Chapter 5

Discussion

We split the discussion of the test results in two parts: first we discuss multi-objective optimization with genetic algorithms, and then the preference model UTA$^{\text{GMS}}$.

5.1 Multi-objective optimization

The multi-objective optimization model included in this thesis consists of two scalarizing methods and two genetic algorithms. According to literature, both scalarizing methods have some drawbacks concerning finding solutions for non-convex regions, and in generating a well-distributed representation of the Pareto front. Therefore, genetic algorithms are used to compare and check whether it is a good representation of the Pareto front.

When comparing the results, both methods show more or less the same shape, but the scalarizing methods are able to find solutions with lower costs for similar CO_2 and PM emissions, i.e. more efficient solutions. However, the results from the genetic algorithms can consists of more alternative solutions (depending on the chosen parameters) and take less computational time. Regards the computational time, for the weighted sum method you need about 5000 runs to obtain the front, which takes about five hours, whereas a genetic algorithm can produce a front in one run, which does can take between two minutes and one hour depending on the amount of generations. In case of NSGA-II, a lower amount of generations produces a well-distributed front, but are not so efficient like the scalarizing methods. To obtain solutions with NSGA-II that are more efficient, similar to the scalarizing methods, an increase of the generations is required. However, in this particular case, the front converges to the first objective if the amount of generations is increased and then it loses their well-spread distribution of the front. However, SPEA2 showed that it maintained the well-spread distribution for a large amount of generations but are still not so efficient compared to the scalarizing methods, whereas NSGA-II is efficient, at least for the first two objectives (costs and CO_2).

The Pareto front obtained by applying genetic algorithms depends on the parameters that are used. However, the obtained front will be similar in each run. By adjusting the initial population size more or fewer solutions can be obtained, but one characteristic of the genetic algorithms is that it produces well-distributed fronts so different parameters will not change much to the fronts presented in this thesis except the amount of generations for the NSGA-II.

When you compare the results of the scalarizing methods with the results of the SPEA2 (and

Table 13: Comparison of MOO methods

Method	No. solutions	Dist. front	Comp. time	Single crit.	Extremes
Weighted sum	(212 + 245)	well	5.5 hours	efficient	close
ε-constraint	(130 + 92)	well	45 min	efficient	close
NSGA-II[a]	(213 + 216)	well/bad	2 min/1 hour	bad/efficient	approximates
SPEA2[b]	(1943 + 1384)	well	2 min/4 hours[c]	bad	approximates

[a] The algorithm generates different results depending on how many generations are used: well-distributed for low amount of generations (but than lacks efficiency), badly distributed for many generations. This also determines the computation time.

[b] The algorithm can also be limited by a bounded archive size, that reduces the amount of solutions and computation time.

[c] A representation can be obtained in two minutes, increase the amount of generations will also increase the number of solutions for the unbounded archive, but does not really improve the front after, e.g., 5000 generations. A bounded archive can reduce the computation time.

NSGA-II in case if few generations are used) you see that the genetic algorithms have problems with finding the solutions with the lowest costs possible for a certain amount of CO_2 and PM emissions. The tables in section 4.4 show that when you compare solutions from the different methods, with CO_2 and PM being more or less the same, the costs in the genetic algorithms are higher. This means that the genetic algorithms do not find the best possible solutions and thus not all solutions are Pareto optimal if you compare them with the real Pareto front.

Both the scalarizing methods and the genetic algorithms showed that if you want to decrease the PM emissions, the costs and CO_2 emissions increase. If you compare the extreme points, you see that when the costs are optimized the CO_2 emissions are near the optimum (minimum) while the PM emissions are near the worst (maximum). Thus optimizing the CO_2 emissions causes a small increase of costs while it also decreases the PM emissions a little. Optimizing the PM emissions will both increase costs and CO_2.

Table 13 presents an overview of the performances of the applied methods to the MOO problem, according to our experience. As the table shows, the scalarizing methods perform well on all criteria except the weighted sum method for the computation time. Both genetic algorithms could improve if they were able to obtain solutions with lower costs like in the weighted sum method and find the extreme solutions. The weighted sum method takes a lot of computation time, while the other methods obtain the fronts relatively fast. However, the weighted sum method finds Pareto optimal solutions, while the genetic algorithms approximates the frontier and thus the solutions are not necessarily Pareto optimal compared with the real Pareto front. Since it is important that the solutions are really close to the real Pareto front, the weighted sum method obtained the best approximation of the Pareto front. When comparing the weighted sum method and epsilon-constraint method, the weighted sum method obtained more efficient solutions for some regions, probably because not the right ε-values are used. However, it is impossible to determine the right ε-values beforehand.

The computation time and amount of solutions obtained by SPEA2 depends on the chosen archive size. The larger the archive, the longer it takes to compute and the more solutions are obtained. It seems however, although the method obtain more solutions, it does not really improve the obtained Pareto front. NSGA-II converges to the first objective if the amount of generations increases. In our case, we used 10^4 generations and you see that the fronts are

Table 14: Alignment objectives; the table shows points near the extremes solutions. It shows how one or both other objectives can improve against a small decrease of the chosen objective

Costs (€1000/yr)	**Δ Costs**	**CO_2** (tn/yr)	**Δ CO_2**	**PM** (Kg/yr)	**Δ PM**
825.47		572.80		27.53	
826.27	0.10%	562.60	-1.78%	23.67	-14.03%
827.33	0.23%	554.71	-3.16%	20.22	-26.57%
827.78	0.28%	554.71	-3.16%	19.01	-30.95%
828.87	0.41%	552.11	-3.61%	16.38	-40.50%
896.58		534.95		11.93	
850.26	-5.17%	534.95	0.00%	13.58	13.77%
894.65	-0.21%	534.98	0.01%	11.55	-3.24%
886.61	-1.11%	537.78	0.53%	12.06	1.06%
858.76	-4.22%	538.37	0.64%	11.68	-2.17%
1039.87		568.18		2.61	
997.98	-4.03%	567.43	-0.13%	2.94	12.86%
988.01	-4.99%	564.59	-0.63%	3.28	25.73%
971.13	-6.61%	564.59	-0.63%	3.65	39.74%
937.78	-9.82%	616.27	8.46%	4.62	77.08%

converged, but become efficient for that region of the front. If a lower amount of generations are used, the obtained front is well-spread but not efficient. Increasing the amount of generations to a larger amount than 10^4 does not improve the results anymore.

This research extends the work of [46] by means of offering various intermediate solutions and combining two options. The results show that there exist many intermediate solutions between the single-objective optimizations by Mallidis et al.. Furthermore, it shows that solutions minimizing the PM emissions outsource the transportation, while solutions that minimize costs or CO_2 lease the transportation. Somewhere around the middle between the minimum and maximum of the PM emissions there exist a few solutions from option A that have less CO_2 emissions, but higher costs, compared to the solutions from option B for a similar level of PM.

The PM emissions are more or less the opposite of the costs and CO_2, costs and CO_2 are more or less aligned. Table 14 shows solutions for the three extreme points of the Pareto front. If you compare these solutions, you see that for a little increase of one objective, there are solutions that reduce both other objectives. This means that even if one objective has a very high priority, there are alternatives nearby that reduce the other two objectives against a small increase of the chosen objective. Especially if you look to the costs, the extreme solution minimizing costs, does only have to increase 0.41% to gain 40.5% loss of PM emissions.

5.2 UTA$^{\mathrm{GMS}}$

We tested whether the preference model UTA$^{\mathrm{GMS}}$ is applicable for problems similar to the case study. In these tests, we tested how many preference statements are needed and the computational time required in the selection process to end up with a single preferred alternative. The tests used random preference statements and each alternative can be chosen once, which prevents the possibility of an inconsistent model. It seems reasonable that when a small amount of preference statements are used per iteration, the DM is capable of applying consistent preference statements. However, for a larger amount of statements it can happen that the transitivity property is violated. Then the preference statements should be revised and the iteration has to start all over again, so if there are many alternatives left, it can take a considerable amount of time. Another point with the random statements is that because each alternative can be chosen once in each iteration, it is not possible to say, e.g., $a \succ b$ and $c \succ a$, because a is already chosen in the same iteration. This means that the latter has to wait until the next iteration before the statement can be added. However, if a real DM is used instead of random preference statements this is not an issue and it would not affect the computation time.

Adding more preference statements per iteration will increase the computation time for that single iteration. This also means that after the iteration, more solutions are necessarily less preferred to another solution. This means that none of these alternatives will be the 'best' alternative (because other solutions are preferred over them), and the total computation time will decrease because there are fewer iterations required. It is reasonable to assume that for n alternatives, $(n-1)$ preference statements are needed. However, if the DM already should know what solution will be his/her preferred one, no preference model is needed at all. Therefore, the DM gets in each iteration a small subset of alternatives, that he/she is capable of expressing preference statements on, and ends up with a single preferred solution.

It is shown in [49] that, even for Pareto sets, the UTA$^{\mathrm{GMS}}$ method is capable of inducing additional (preference) relations from a single preference statement. However, in our case this occurs only a few times while many statements are used. Thus given a preference statement in a reference set ($x \succ y \in A^R$), it would not induce any additional relations for a pair of alternatives in the whole set (such as, $w \succ z \in A$). Therefore, you need $(n-1)$ preference statements to obtain a complete preorder (a 'ranking' problem), and also $(n-1)$ preference statements to choose the 'best' alternative (a 'choosing' problem). Thus because it will not induce any other relations than given by the preference statements and the (long) computation time for large samples, the UTA$^{\mathrm{GMS}}$ method is not appropriate to use for the case described in this study.

Chapter 6

Conclusions and future work

The study was set out to investigate whether a preference model like UTA$^{\text{GMS}}$ is applicable for cases like the one described in Chapter 4. In order to do this, we first had to deal with a multi-objective optimization problem. To this intent, we applied two 'simple' scalarizing methods and a more complex method: genetic algorithms. Both scalarizing methods have some issues in obtaining a well-distributed representation of the Pareto front, therefore, two genetic algorithms are used to compare the obtained fronts, which allows us to see whether the scalarizing methods are sufficient for obtaining a front or whether some essential solutions are missing. Both methods were compared in order to see differences in the impact on cost and design in green logistics. Then the obtained representation of the Pareto front has used to test whether the UTA$^{\text{GMS}}$ is applicable to the case. The latter has not been done yet in existing literature (as far as we know). We tested the method by estimating the time required to find one single 'best' solution and by counting the number of preference statements that are required in order to find the 'best' solution.

6.1 Conclusions

Our main research question was: *Can we aid the DM to design a supply chain in green logistics, by representing alternatives obtained with genetic algorithm and thereafter apply the UTA^{GMS} method?* In order to answer this question, it is decomposed into subquestions. These subquestions have been answered in previous chapters, we will now summarize each of them.

1. *How different is the Pareto front obtained by a genetic algorithm (NSGA-II/SPEA2) related to the weighted sum method?*
   The largest difference between these types of methods is the efficiency in the obtained solutions. The Pareto front representation obtained with the scalarizing methods are more efficient than the ones obtained with the genetic algorithms, i.e., they are closer to the real Pareto front. The latter might win in computational terms, but in the end is efficiency often the most important criteria for selecting a supply chain design. The weighted sum method obtains an acceptable amount of alternatives and produces a good representation of the Pareto front. It seems the method does not suffer from missing solutions for non-convex regions since the genetic algorithms were not able to improve the obtained fronts.

Therefore, we would recommend to use the scalarizing approach for this particular case to obtain a well-spread Pareto front with efficient solutions and probably close to the real Pareto front.

2. *How many preference statements do we need in order to obtain a single (best) solution?* To this question, a short answer can be given: for n alternatives, you can assume that it needs $(n-1)$ preference statements. During our tests, it occasionally happened that more solutions could be removed than the number of preference statements given. But this did happen only a few times, so you can assume that UTA$^{\text{GMS}}$ needs $(n-1)$ preference statements until the DM's single preferred solution remains.

3. *How applicable is the UTA^{GMS} method to this specific multi-objective optimization model?* As we have seen takes it really long if the possible solution set is larger than, e.g., fifty alternatives. The Pareto fronts we have obtained consist of 190-250 possible solutions, thus in such cases one can say that UTA$^{\text{GMS}}$ is not applicable to this case. However, it is shown that for smaller solution sets it is able in a appropriate time to find the DM's 'best' solution. Thus either the genetic algorithm's parameters can be adjusted to give a representation of the Pareto front with a smaller set of possible solution or some kind of pre-selection mechanism is required.

Finally, we are able to answer the research question: *Can we aid the DM to design a supply chain in green logistics, by representing alternatives obtained with genetic algorithm and thereafter apply the UTA^{GMS} method?*

We have seen that both genetic algorithms were capable of presenting a representation of the Pareto front, i.e., give the DM a set of possible configurations for the design of the supply chain. We showed that these configurations can differ with respect to the number of distribution centers or entry points that are used, and we showed some payoff tables in which the difference were listed with respect to the objectives. Related to the scalarizing methods, the genetic algorithms produced a larger set of possible solutions, had a close approximation of the extreme points, but the solutions are not so efficient as the ones obtained with the scalarizing methods. Thus it is better to present the DM a set of alternatives resulting from the scalarizing methods, in particular the weighted sum method.

The obtained results have been used for the preference model. As we have seen the UTA$^{\text{GMS}}$ method is appropriate when the set of possible solutions is small, e.g., a maximum of 50 possible solutions. It can be used for larger sets, but then the DM needs a lot of patience. Thus the final answer to the research question is no, because it does not induce any other preference relation than the ones provided by the DM.

There is already a lot of work done in the topic of multi-critera decision analysis (MCDA), but none of them have assessed whether the preference models have been applicable to real-world decision problems in green logistics. This research is a contribution to this field in the sense that it assessed both the computational time that is required to use such a model, as well as the number of preference statements required for a DM to arrive at his/her final solution. The preference model can also be used to obtain a complete preorder. However, we think the

DM is only interested in what solution would be his/her preferred one and not interested in the complete preorder.

6.2 Future research

This research uncovered some directions for future research. One feature of the UTA^{GMS} method is that it allows to build a complete preorder of the possible solution set. In this research, however, we only include the choice problem, i.e., aid the DM to find his/her most preferred solution. Future research should assess on the number of preference statements needed for obtaining a complete preorder, and the corresponding computation time.

To this end, one can also investigate how often these preference statements lead to an inconsistent preference model. In our case it was not possible to get an inconsistent model, because we only allow one alternative to be chosen once per iteration. Especially when a large amount of preference statements are applied, to obtain a complete preorder, there is a chance that the statements are inconsistent.

Furthermore, one can think about how the computational time could be reduced for a large set of possible solutions. Now, for each alternative are LP problems to be solved to determine whether the alternative is necessary better than other ones, thus the relation needs to be computed in both directions. It might possible that when you check whether a is necessary better than b, it also involves whether b is necessary better than a. Additional improvements might be obtained when the preference model would require fewer statements to obtain a final solution.

Moreover, the results of NSGA-II can probably be improved by solving the issue that it is converging to the first objective. Also, maybe SPEA2 can be improved to some extend such that it is able to find solutions that are closer to the real Pareto front.

Bibliography

[1] F. Altiparmak, M. Gen, L. Lin, and T. Paksoy. A genetic algorithm approach for multi-objective optimization of supply chain networks. *Computers & Industrial Engineering*, 51 (1):196–215, 2006.

[2] S. Alumur and B. Y. Kara. A new model for the hazardous waste location-routing problem. *Computers and Operations Research*, 34(5):1406–1423, 2007.

[3] S. Angilella, S. Greco, and B. Matarazzo. Non-additive robust ordinal regression: A multiple criteria decision model based on the choquet integral. *European Journal of Operational Research*, 201(1):277–288, 2010.

[4] C. A. Bana e Costa and J. C. Vansnick. MACBETH — An interactive path towards the construction of cardinal value functions. *International Transactions in Operational Research*, 1(4):489–500, 1994.

[5] C. A. Bana e Costa and J. C. Vansnick. Applications of the MACBETH Approach in the Framework of an Additive Aggregation Model. *Journal of multi-criteria decision analysis*, 6(2):107–114, 1997.

[6] J. Barzilai. Preference Function Modeling: The Mathematical Foundations of Decision Theory. In M. Ehrgott, José, R. Figueira, and S. Greco, editors, *Trends in Multiple Criteria Decision Analysis*, pages 57–86. Springer, 2010.

[7] D. Beasley, D. R. Bull, and R. R. Martin. An overview of genetic algorithms: Part 1, fundamentals, 1993.

[8] L. Boer, F. Brouwer, and H. Essen. STREAM, Study for Transport Emissions of all Modes. Version 2, Rapport, CE Delft September 2008 (in Dutch). `http://www.ce.nl/publicatie/stream%3A_study_on_trasnport_emissions_of_all_nodes/832`, 2008.

[9] D. Bouyssou and P. Vincke. Introduction to topics on preference modelling. *Annals of Operations Research*, 80(1):i–xiv, 2008.

[10] J. Branke, S. Greco, R. Słowiński, and P. Zielniewicz. Interactive evolutionary multiobjective optimization driven by robust ordinal regression. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, 58(3):347–358, 2010.

[11] J.-P. Brans and B. Mareschal. PROMETHEE Methods. In J. Figueira, S. Greco, and M. Ehrogott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, volume 78 of *International Series in Operations Research & Management Science*, pages 163–186. Springer, 2005.

[12] S. Cahon, N. Melab, and E.-G. Talbi. ParadisEO: A Framework for the Reusable Design of Parallel and Distributed Metaheuristics. *Journal of Heuristics*, 10(3):357–380, 2004.

[13] I. Das and J. E. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. *Structural and Multidisciplinary Optimization*, 14(1):63–69, 1997.

[14] K. Deb. Multi-objective optimization. In *Search Methodologies*, pages 273–316. Springer US, 2005.

[15] K. Deb and J. Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 635–642. ACM, 2006.

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computations*, 6(2): 182–197, 2002.

[17] S. Ebert. Literature review: Inland Navigation and Emissions Efficiency. WWF International Danube-Carpathian Program. `http://assets.panda.org/downloads/wwf_iwt_emissions_lit_review.pdf`, 2005.

[18] M. Ehrgott. Multiobjective Optimization. *AI Magazine*, 29(4):47–57, 2008.

[19] EU. Fright transport logistcs action plan. `http://ec.europa.eu/transport/logistics/freight_logistics_action_plan/doc/action_plan/2007_com_logistics_action_plan_en.pdf`, 2007.

[20] EU. Greening transport. `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2008:0433:FIN:EN:PDF`, 2008.

[21] EU. Strategy for the internalization of external costs. `http://ec.europa.eu/transport/greening/doc/costs/2008_07_greening_transport_cost_communication_en.pdf`, 2008.

[22] EU. A sustainable future for transport: Towards an integrated, technology led and user friendly system. `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2009:0279:FIN:EN:PDF`, 2009.

[23] J. Figueira, V. Mousseau, and B. Roy. ELECTRE Methods. In J. Figueira, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 170–190. Springer, 2005.

[24] J. Figueira, A. Liefooghe, E.-G. Talbi, and A. Wierzbicki. A parallel multiple reference point approach for multi-objective optimization. *European Journal of Operational Research*, 205 (2):390–400, 2010.

[25] J. R. Figueira, S. Greco, and R. Słowiński. Building a set of additive value functions representing a reference preorder and intensities of preference: Grip method. *European Journal of Operational Research*, 195(2):460–486, 2009.

[26] C. M. Fonseca and P. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.

[27] M. Gen and R. Cheng. *Genetic algorithm and engineering design*. Wiley, 1997.

[28] P. Goodwin and G. Wright. *Decision Analysis for management Judgment*. Wiley, 3th edition, 2004.

[29] S. Greco, V. Mousseau, and R. Słowiński. Ordinal regression revisited: Multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, 191(2):415–435, 2008.

[30] S. Greco, V. Mousseau, and R. Słowiński. Multiple criteria sorting with a set of additive value functions. *European Journal of Operational Research*, 207(3):1455–1470, 2010.

[31] S. Greco, M. Kadziński, V. Mousseau, and R. Słowiński. ELECTRE$^{\mathrm{GKMS}}$: Robust ordinal regression for outranking methods. *European Journal of Operational Research*, 214(1):118–135, 2011.

[32] S. Greco, M. Kadziński, V. Mousseau, and R. Słowiński. Robust ordinal regression for multiple criteria group decision: UTAGMS-GROUP and UTADISGMS-GROUP. *Decision Support Systems*, 52(3):549–561, 2012.

[33] P. Hajela and C. Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural and Multidisciplinary Optimization*, 4:99–107, 1992.

[34] I. Harris, M. Naim, A. Palmer, A. Potter, and C. Mumford. Assessing the impact of cost optimization based on infrastructure modelling on co2 emissions. *International Journal of Production Economics*, 131(1):313–321, 2011.

[35] J. Horn, J. Horn, N. Nafpliotis, N. Nafpliotis, D. E. Goldberg, and D. E. Goldberg. A Niched Pareto Genetic Algorithm for Multiobjective Optimization. In *In Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, pages 82–87, 1994.

[36] Ioannis and Giannikos. A multiobjective programming model for locating treatment sites and routing hazardous wastes. *European Journal of Operational Research*, 104(2):333–342, 1998.

[37] E. Jacquet-Lagreze and J. Siskos. Assessing a set of additive utility functions for multicriteria decision-making, the UTA method. *European Journal of Operational Research*, 10(2): 151–164, 1982.

[38] M. Kadziński, S. Greco, and R. Słowiński. Extreme ranking analysis in robust ordinal regression. *Omega*, 40(4):488–501, 2012.

[39] R. L. Keeney and H. Raiffa. *Decisions with multiple objectives: preferences and value tradeoffs*. Cambridge University Press, 1993.

[40] I. Kim and O. de Weck. Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. *Structural and Multidisciplinary Optimization*, 29(2):149–158, 2005.

[41] I. Kim and O. de Weck. Adaptive weighted sum method for multiobjective optimization: a new method for Pareto front generation. *Structural and Multidisciplinary Optimization*, 31(2):105–116, 2006.

[42] J. Knowles and D. Corne. The pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization. In *Proceedings of the 1999 Congress on Evolutionary Computation (CEC'99)*, pages 98–105, 1999.

[43] M. Laumanns, L. Thiele, and E. Zitzler. An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method. *European Journal of Operational Research*, 169(3):932–942, 2006.

[44] A. Liefooghe, M. Basseur, L. Jourdan, and E.-G. Talbi. ParadisEO-MOEO: A Framework for Evolutionary Multi-objective Optimization. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, editors, *Evolutionary Multi-Criterion Optimization*, volume 4403, pages 386–400. Springer Berlin / Heidelberg, 2007.

[45] I. Mallidis, R. Dekker, and D. Vlachos. Greening supply chains: Impact on cost and design. Econometric Institute Report EI 2010-39b, Erasmus University Rotterdam, Econometric Institute, Aug. 2010.

[46] I. Mallidis, R. Dekker, and D. Vlachos. The impact of greening on supply chain design and cost: a case for a developing region. *Journal of Transport Geography*, 22:118–128, 2012.

[47] G. Mavrotas. Effective implementation of the ε-constraint method in multi-objective mathematical programming problems. *Applied Mathematics and Computation*, 213(2):455–465, 2009.

[48] K. Miettinen and M. M. Mäkelä. On scalarizing functions in multiobjective optimization. *OR Spectrum*, 24(2):193–213, 2002.

[49] Mousseau. Preference elicitation for MCDA Robust elicitation of ranking model. `http://www.lamsade.dauphine.fr/~gold/Cours%20et%20pr%e9sentations/Mousseau3.pdf`, 2008.

[50] A. K. Nema and S. Gupta. Optimization of regional hazardous waste management systems: an improved formulation. *Waste Management*, 19(7):441–451, 1999.

[51] J. Quariguasi Frota Neto, J. Bloemhof-Ruwaard, J. van Nunen, and E. van Hoek. Designing and evaluating sustainable logistics networks. *International Journal of Production Economics*, 111(2):195–208, 2008.

[52] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2011. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

[53] B. Roy. The outranking approach and the foundations of ELECTRE methods. *Theory and Decision*, 31(1):49–73, 1991.

[54] B. Roy and D. Bouyssou. *Aide multicritère à la décision : méthodes et cas*. Economica, Paris, 1993.

[55] A. Sbihi and R. W. Eglese. Combinatorial optimization and Green Logistics. *4QR: A Quarterly Journal of Operations Research*, 5(2):99–116, 2007.

[56] J. D. Schaffer. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 93–100. L. Erlbaum Associates Inc., 1985.

[57] L. Simpson. Do decision makers know what they prefer?: Mavt and electre ii. *The Journal of Operational Research Society*, 47(7):919–929, 1996.

[58] Y. Siskos. Analyse de systèmes de décision multicritère en univers aléatoire. *Foundations of Control Engineering*, 10(3–4):193–212, 1983.

[59] Y. Siskos and D. Yannacopoulos. UTASTAR: An ordinal regression method for building additive value functions. *Investiga cão Operacional*, 5(1):39–53, 1985.

[60] Y. Siskos, E. Grigoroudis, and N. Matsatsinis. UTA Methods. In J. Figueira, S. Greco, and M. Ehrgott, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 297–334. Springer, 2005.

[61] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[62] A. Syarif, Y. Yun, and M. Gen. Study on multi-stage logistic chain network: a spanning tree-based genetic algorithm approach. *Computers & Industrial Engineering*, 43(1):299–314, 2002.

[63] L. Thiele, K. Miettinen, P. J. Korhonen, and J. Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evolutionary Computation*, 17(3):411–436, 2009.

[64] L. Zheng and J. Zhang. Research on Green Logistics System Based on Circular Economy. *Asian Social Science*, 6(11):116–119, 2010.

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

[66] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, 2001.

Appendix A: R script

This file is also available at github: `https://github.com/CornePlas/MOOSCN-R`.

```
file_GA <- "OptionA_nsgaii.txt";

# load required libraries
library("ror");
library("sampling");
library("chron");
library("MASS");

# read data
data <- as.matrix(read.csv(file_GA, header=FALSE, sep=";"));
data_adj <- 1/data;

# function to find one single solution,
# it uses random preference statements to filter alternatives
findSol <- function(d, s){
  pref.s <- c();
  size <- nrow(d);
  statements <- 0;
  iterations <- 0;
  maxTime <- 0;

  # create loop to end with one variable
  while( (is.matrix(d)) && (length(d[,1] > 0)) ){
    # amount divided by two are the number of statements, thus s
    pref.s <- s * 2;
    selection <- matrix(numeric(1), length(d[,1]), 1);


    # determine amount of preference statements
    x <- 1:length(d[,1]);

    while (pref.s > length(d[,1]) ||
```

```
                (pref.s == length(d[,1]) && !(pref.s%%2==0))){
        # if the remaining alternatives is smaller than the
        # number of alternatives, less statements are needed
        pref.s <- round((pref.s/2)/2)*2;
    }

    # display some output
    cat(format(Sys.time(), "%H:%M;"), "␣iteration:␣",iterations,",
␣size:␣",nrow(d),"/",size,",␣statements:",(pref.s/2));

    cat(format(Sys.time(), "%H:%M;"), "␣iteration:␣",iterations,",
␣size:␣",nrow(d),"/",size,",␣statements:",(pref.s/2),
 file="output_small.txt", append=TRUE);

    # count no. statements
    statements <- statements + (pref.s / 2);
    # create random pref. statements
    preferences <- matrix(sample(x, pref.s, replace=FALSE),
                                          ncol=2, byrow=TRUE);
    # calc necessary relations
    timeIt <- system.time(ror <- utagms(d, preferences,
                          necessary=TRUE, strictVF=TRUE))[1];

    # show some output
    cat(",␣time:",timeIt,'\n');
    cat(",␣time:",timeIt,'\n',file="output.txt",append=TRUE);
    for (i in 1:length(ror[,1])){
    # filter all alternatives that cannot be the preferred one
      for (j in 1:length(ror[1,])){
        if (i != j && ror[i,j])
          selection[j] = 1;
      }
    }

    if (timeIt > maxTime){
      maxTime <- timeIt;
    }

    d <- d[selection==0,]; # remove 'dominated' solutions
    iterations <- iterations + 1;
  }
  return(c(d, statements, iterations, maxTime));
```

```
}

## use sample
sample_size <- c(10, 20, 30, 40, 50, 60);
no_statements <- c(5,10,20);
results <- matrix(numeric(1), (length(sample_size)*length
(no_statements)), 6)
for(j in 1:length(sample_size)){
  for (i in 1:length(no_statements)){
    if (sample_size[j] >= (no_statements[i]*2)){
      sample_data <- srswor(sample_size[j], nrow(data));
      sample_set <- as.matrix(data_adj[sample_data==1,1:3]);
      sys <- system.time(
                x <- findSol(sample_set, no_statements[i]));
      results[j,] <- c(sample_size[j], no_statements[i],
                       sys[1], x[4], x[5], x[6]);
        # no. alternatives used, time for calc, no. statements,
        # no. iterations, (fixed) statements p/iteration
      cat(sample_size[j],no_statements[i],sys[1],x[4],x[5],
                x[6],'\n',file="results.txt",sep=",",append=TRUE);
      cat(results[j,],"\n");
    }
  }
}

print(results)
write.matrix(results, file="results_all.csv", sep=";", 1024);

cat('',file="output_large.txt");
sample_large <- c(75, 100, 125, 150, 175, 200);
statements <- 25;
results <- matrix(numeric(1), 6, 3);
for (i in 1:length(sample_large)){
  sample_data <- srswor(sample_large[i], nrow(data));
  sample_set <- as.matrix(data_adj[sample_data==1,1:3]);
  preferences <- matrix(sample(1:nrow(sample_set), statements,
                        replace=FALSE), ncol=2, byrow=TRUE);
  sys <- system.time(ror <- utagms(sample_set, preferences,
                                   necessary=TRUE, strictVF=TRUE))
  results[i,] <- c(sample_large[i], statements, sys[1])
  cat(results[i,],"\n");
  cat(results[i,],'\n', file="output_large.txt", sep=",",
```

```
                                                    append=TRUE);
}

print(results)
write.matrix(results, file="results_large.csv", sep=";", 1024);
```

Appendix B: Data

The data is also available at github:
`https://github.com/CornePlas/MOOSCN/tree/master/data`

Table 15: Demand

rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
22	18	17	19	19	42	19	11	7	10	9	9	11	15	19

Table 16: Warehouse costs (per item)

dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
96	96	96	96	96	96	96	96	72	96	96	96	96	96	96	120

Table 17: Costs/Emissions from Major Loading Point to EP

	ep1	**ep2**	**ep3**
Costs	2317	2621	2607
CO_2	1963	2027	2040
PM	5119	5284	5317

Option A

Table 18: dcrm-dt1-cost

	rm1	rm2	rm3	rm4	rm5	rm6	rm7	rm8	rm9	rm10	rm11	rm12	rm13	rm14	rm15
dc1	220	2479	2678	3194	2659	2770	2567	2616	3440	2837	2897	3278	3376	3407	3161
dc2	2479	220	2518	2661	2132	2260	2747	2466	3296	2655	2409	2786	2882	2942	2903
dc3	2678	2518	220	2800	2665	2721	3272	3012	3866	3182	2860	3040	3198	3364	3440
dc4	3194	2661	2800	220	2569	2405	3411	2835	3493	3053	2444	2253	2505	2948	3061
dc5	2659	2132	2665	2569	220	2128	2940	2466	3387	2653	3167	2815	3040	3557	3038
dc6	2770	2260	2721	2405	2128	220	2928	2403	3118	2591	2093	2469	2567	2589	2678
dc7	2567	2747	3272	3411	2940	2928	220	2567	3092	2546	3055	3428	3512	3303	2923
dc8	2616	2467	3012	2835	2466	2403	2567	220	2823	2135	2401	2903	2895	2673	2426
dc9	3440	3296	3866	3493	3387	3118	3092	2823	220	2635	3036	3233	3159	2659	2387
dc10	2837	2655	3182	3053	2653	2591	2546	2135	2635	220	2614	2966	3145	2649	2313
dc11	2538	2409	2860	2444	3167	2093	3055	2401	3036	2614	220	2321	2426	2448	2538
dc12	3278	2786	3040	2253	2815	2469	3428	2903	3233	2966	2321	220	2196	2665	2891
dc13	3376	2882	3198	2505	3040	2567	3512	2895	3159	3145	2426	2190	220	2475	2731
dc14	3419	2942	3364	2948	3557	2589	3303	2673	2659	2649	2448	2665	2475	220	2221
dc15	3161	2903	3440	3061	3038	2678	2923	2426	2387	2313	2538	2891	2731	2221	220
dc16	3737	3506	4088	3655	4284	3299	3387	3030	2389	2850	3176	3337	3094	2641	2509

Table 19: dcrm-dt1-pm

	rm1	rm2	rm3	rm4	rm5	rm6	rm7	rm8	rm9	rm10	rm11	rm12	rm13	rm14	rm15
dc1	6.2	167.6	230.4	393	224.2	259.3	195.3	210.7	470.6	2176.3	299.4	419.5	450.3	460.2	382.5
dc2	167.6	6.2	179.9	224.8	57.9	98.6	251.9	163.2	425	223	145.4	264.3	294.4	313.5	301.2
dc3	230.4	179.9	6.2	268.6	226.1	243.9	417.6	335.7	604.9	389.3	287.7	344.3	394.2	446.6	470.6
dc4	393	224.8	268.6	6.2	195.9	144.1	461.4	279.7	487.3	348.7	156.5	96.1	175.6	315.4	351.1
dc5	224.2	57.9	226.1	195.9	6.2	56.7	312.9	163.2	454	222.4	384.4	273.5	344.3	507.6	343.7
dc6	259.3	98.6	243.9	144.1	56.7	6.2	309.2	143.5	369	202.7	45.6	164.5	195.3	202	230.4
dc7	195.3	251.9	417.6	461.4	312.9	309.2	6.2	195.3	361	188.5	349.3	466.9	493.4	427.5	307.4
dc8	210.7	163.9	335.7	279.7	163.2	143.5	195.3	6.2	276	59.1	142.9	301.2	298.8	228.5	150.9
dc9	470.6	425	604.9	487.3	454	369	361	276	6.2	216.8	343.1	405.3	381.9	224.2	138.6
dc10	2176.3	223	389.3	348.7	222.4	202.7	188.5	59.1	216.8	6.2	210.1	320.9	377.6	221.1	115.2
dc11	186	145.4	287.7	156.5	384.4	45.6	349.3	142.9	343.1	210.1	6.2	117.7	150.9	157.7	186
dc12	419.5	264.3	344.3	96.1	273.5	164.5	466.9	301.2	405.3	320.9	117.7	6.2	78.2	226.1	297.5
dc13	450.3	294.4	394.2	175.6	344.3	195.3	493.4	298.8	381.9	377.6	150.9	76.4	6.2	166.3	247.1
dc14	463.8	313.5	446.6	315.4	507.6	202	427.5	228.5	224.2	221.1	157.7	226.1	166.3	6.2	86.2
dc15	382.5	301.2	470.6	351.1	343.7	230.4	307.4	150.9	138.6	115.2	186	297.5	247	86.2	6.2
dc16	564.3	491.6	675.1	538.4	736.7	426.3	454	341.3	139.2	284.6	387.5	438	361.6	218.7	176.8

Table 20: dcrm-dt2-cost

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	238	2685	2901	3459	2880	3000	2780	2833	3726	3072	3138	3550	3656	3690	3423
dc2	2685	238	2727	2882	2309	2448	2975	2670	3569	2875	2609	3017	3121	3186	3144
dc3	2901	2727	238	3032	2886	2947	3544	3262	4187	3446	3097	3292	3463	3643	3726
dc4	3459	2882	3032	238	2782	2605	3694	3070	3783	3307	2647	2440	2712	3193	3315
dc5	2880	2309	2886	2782	238	2304	3184	2670	3669	2873	3430	3049	3292	3853	3290
dc6	3000	2448	2947	2605	2304	238	3171	2603	3377	2806	2266	2674	2780	2803	2901
dc7	2780	2975	3544	3694	3184	3171	238	2780	3349	2757	3309	3713	3804	3578	3165
dc8	2833	2672	3262	3070	2670	2603	2780	238	3057	2313	2600	3144	3136	2894	2628
dc9	3726	3569	4187	3783	3669	3377	3349	3057	238	2854	3288	3501	3421	2880	2586
dc10	3072	2875	3446	3307	2873	2806	2757	2313	2854	238	2831	3212	3406	2869	2505
dc11	2748	2609	3097	2647	3430	2266	3309	2600	3288	2831	238	2514	2628	2651	2748
dc12	3550	3017	3292	2440	3049	2674	3713	3144	3501	3212	2514	238	2378	2886	3131
dc13	3656	3121	3463	2712	3292	2780	3804	3136	3421	3406	2628	2372	238	2681	2958
dc14	3702	3186	3643	3193	3853	2803	3578	2894	2880	2869	2651	2886	2681	238	2406
dc15	3423	3144	3726	3315	3290	2901	3165	2628	2586	2505	2748	3131	2958	2406	238
dc16	4047	3798	4428	3958	4639	3573	3669	3281	2588	3087	3440	3614	3351	2861	2717

Table 21: dcrm-dt2-pm

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	1.2	33.5	46.1	78.6	44.8	51.9	39.1	42.1	94.1	56.1	59.9	83.9	90.1	92	76.5
dc2	33.5	1.2	36	45	11.6	19.7	50.4	32.6	85	44.6	29.1	52.9	58.9	62.7	60.2
dc3	46.1	36	1.2	53.7	45.2	48.8	83.5	67.1	121	77.9	57.5	68.9	78.8	89.3	94.1
dc4	78.6	45	53.7	1.2	39.2	28.8	92.3	55.9	97.5	69.7	31.3	19.2	35.1	63.1	70.2
dc5	44.8	11.6	45.2	39.2	1.2	11.3	62.6	32.6	90.8	44.5	76.9	54.7	68.9	101.5	68.7
dc6	51.9	19.7	48.8	28.8	11.3	1.2	61.8	28.7	73.8	40.5	9.1	32.9	39.1	40.4	46.1
dc7	39.1	50.4	83.5	92.3	62.6	61.8	1.2	39.1	72.2	37.7	69.9	93.4	98.7	85.5	61.5
dc8	42.1	32.8	67.1	55.9	32.6	28.7	39.1	1.2	55.2	11.8	28.6	60.2	59.8	45.7	30.2
dc9	94.1	85	121	97.5	90.8	73.8	72.2	55.2	1.2	43.4	68.6	81.1	76.4	44.8	27.7
dc10	56.1	44.6	77.9	69.7	44.5	40.5	37.7	11.8	43.4	1.2	42	64.2	75.5	44.2	23
dc11	37.2	29.1	57.5	31.3	76.9	9.1	69.9	28.6	68.6	42	1.2	23.5	30.2	31.5	37.2
dc12	83.9	52.9	68.9	19.2	54.7	32.9	93.4	60.2	81.1	64.2	23.5	1.2	15.6	45.2	59.5
dc13	90.1	58.9	78.8	35.1	68.9	39.1	98.7	59.8	76.4	75.5	30.2	15.3	1.2	33.3	49.4
dc14	92.8	62.7	89.3	63.1	101.5	40.4	85.5	45.7	44.8	44.2	31.5	45.2	33.3	1.2	17.2
dc15	76.5	60.2	94.1	70.2	68.7	46.1	61.5	30.2	27.7	23	37.2	59.5	49.4	17.2	1.2
dc16	112.9	98.3	135	107.7	147.3	85.3	90.8	68.3	27.8	56.9	77.5	87.6	72.3	43.7	35.4

Table 22: dcrm-dt3-cost

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	257	2896	3128	3731	3106	3236	2998	3055	4018	3313	3384	3829	3943	3979	3692
dc2	2896	257	2941	3108	2490	2640	3208	2880	3849	3101	2814	3254	3366	3436	3391
dc3	3128	2941	257	3270	3112	3179	3822	3518	4515	3717	3341	3550	3735	3929	4018
dc4	3731	3108	3270	257	3001	2809	3984	3311	4080	3566	2855	2631	2925	3443	3575
dc5	3106	2490	3112	3001	257	2485	3434	2880	3956	3099	3699	3288	3550	4155	3548
dc6	3236	2640	3179	2809	2485	257	3420	2807	3642	3026	2444	2884	2998	3023	3128
dc7	2998	3208	3822	3984	3434	3420	257	2998	3612	2973	3569	4004	4102	3858	3414
dc8	3055	2882	3518	3311	2880	2807	2998	257	3297	2494	2804	3391	3382	3122	2834
dc9	4018	3849	4515	4080	3956	3642	3612	3297	257	3078	3546	3776	3690	3106	2789
dc10	3313	3101	3717	3566	3099	3026	2973	2494	3078	257	3053	3464	3674	3094	2702
dc11	2964	2814	3341	2855	3699	2444	3569	2804	3546	3053	257	2711	2834	2859	2964
dc12	3829	3254	3550	2631	3288	2884	4004	3391	3776	3464	2711	257	2565	3112	3377
dc13	3943	3366	3735	2925	3550	2998	4102	3382	3690	3674	2834	2558	257	2891	3190
dc14	3993	3436	3929	3443	4155	3023	3858	3122	3106	3094	2859	3112	2891	257	2595
dc15	3692	3391	4018	3575	3548	3128	3414	2834	2789	2702	2964	3377	3190	2595	257
dc16	4365	4096	4775	4269	5003	3854	3956	3539	2791	3329	3710	3897	3614	3085	2930

Table 23: dcrm-dt3-pm

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	1.2	33.5	46.1	78.6	44.8	51.9	39.1	42.1	94.1	56.1	59.9	83.9	90.1	92	76.5
dc2	33.5	1.2	36	45	11.6	19.7	50.4	32.6	85	44.6	29.1	52.9	58.9	62.7	60.2
dc3	46.1	36	1.2	53.7	45.2	48.8	83.5	67.1	121	77.9	57.5	68.9	78.8	89.3	94.1
dc4	78.6	45	53.7	1.2	39.2	28.8	92.3	55.9	97.5	69.7	31.3	19.2	35.1	63.1	70.2
dc5	44.8	11.6	45.2	39.2	1.2	11.3	62.6	32.6	90.8	44.5	76.9	54.7	68.9	101.5	68.7
dc6	51.9	19.7	48.8	28.8	11.3	1.2	61.8	28.7	73.8	40.5	9.1	32.9	39.1	40.4	46.1
dc7	39.1	50.4	83.5	92.3	62.6	61.8	1.2	39.1	72.2	37.7	69.9	93.4	98.7	85.5	61.5
dc8	42.1	32.8	67.1	55.9	32.6	28.7	39.1	1.2	55.2	11.8	28.6	60.2	59.8	45.7	30.2
dc9	94.1	85	121	97.5	90.8	73.8	72.2	55.2	1.2	43.4	68.6	81.1	76.4	44.8	27.7
dc10	56.1	44.6	77.9	69.7	44.5	40.5	37.7	11.8	43.4	1.2	42	64.2	75.5	44.2	23
dc11	37.2	29.1	57.5	31.3	76.9	9.1	69.9	28.6	68.6	42	1.2	23.5	30.2	31.5	37.2
dc12	83.9	52.9	68.9	19.2	54.7	32.9	93.4	60.2	81.1	64.2	23.5	1.2	15.6	45.2	59.5
dc13	90.1	58.9	78.8	35.1	68.9	39.1	98.7	59.8	76.4	75.5	30.2	15.3	1.2	33.3	49.4
dc14	92.8	62.7	89.3	63.1	101.5	40.4	85.5	45.7	44.8	44.2	31.5	45.2	33.3	1.2	17.2
dc15	76.5	60.2	94.1	70.2	68.7	46.1	61.5	30.2	27.7	23	37.2	59.5	49.4	17.2	1.2
dc16	112.9	98.3	135	107.7	147.3	85.3	90.8	68.3	27.8	56.9	77.5	87.6	72.3	43.7	35.4

Table 24: dcrm-dt4-cost

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	385	4339	4687	5590	4653	4848	4492	4578	6020	4964	5070	5736	5907	5962	5531
dc2	4339	385	4407	4656	3730	3956	4807	4315	5767	4646	4216	4875	5043	5149	5080
dc3	4687	4407	385	4899	4663	4762	5726	5272	6765	5569	5005	5320	5596	5887	6020
dc4	5590	4656	4899	385	4496	4209	5969	4961	6112	5343	4277	3942	4383	5159	5357
dc5	4653	3730	4663	4496	385	3723	5145	4315	5928	4643	5542	4926	5320	6225	5316
dc6	4848	3956	4762	4209	3723	385	5125	4205	5456	4533	3662	4322	4492	4530	4687
dc7	4492	4807	5726	5969	5145	5125	385	4492	5412	4455	5347	6000	6147	5781	5114
dc8	4578	4318	5272	4961	4315	4205	4492	385	4940	3737	4202	5080	5067	4677	4246
dc9	6020	5767	6765	6112	5928	5456	5412	4940	385	4612	5313	5658	5528	4653	4178
dc10	4964	4646	5569	5343	4643	4533	4455	3737	4612	385	4574	5190	5504	4636	4048
dc11	4441	4216	5005	4277	5542	3662	5347	4202	5313	4574	385	4062	4246	4284	4441
dc12	5736	4875	5320	3942	4926	4322	6000	5080	5658	5190	4062	385	3843	4663	5060
dc13	5907	5043	5596	4383	5320	4492	6147	5067	5528	5504	4246	3833	385	4332	4780
dc14	5983	5149	5887	5159	6225	4530	5781	4677	4653	4636	4284	4663	4332	385	3887
dc15	5531	5080	6020	5357	5316	4687	5114	4246	4178	4048	4441	5060	4780	3887	385
dc16	6540	6136	7155	6396	7497	5774	5928	5302	4181	4988	5559	5839	5415	4622	4390

Table 25: dcrm-dt4-pm

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	0.6	16.8	23	39.3	22.4	25.9	19.5	21.1	47.1	28	29.9	41.9	45	46	38.3
dc2	16.8	0.6	18	22.5	5.8	9.9	25.2	16.3	42.5	22.3	14.5	26.4	29.4	31.4	30.1
dc3	23	18	0.6	26.9	22.6	24.4	41.8	33.6	60.5	38.9	28.8	34.4	39.4	44.7	47.1
dc4	39.3	22.5	26.9	0.6	19.6	14.4	46.1	28	48.7	34.9	15.6	9.6	17.6	31.5	35.1
dc5	22.4	5.8	22.6	19.6	0.6	5.7	31.3	16.3	45.4	22.2	38.4	27.4	34.4	50.8	34.4
dc6	25.9	9.9	24.4	14.4	5.7	0.6	30.9	14.4	36.9	20.3	4.6	16.4	19.5	20.2	23
dc7	19.5	25.2	41.8	46.1	31.3	30.9	0.6	19.5	36.1	18.8	34.9	46.7	49.3	42.8	30.7
dc8	21.1	16.4	33.6	28	16.3	14.4	19.5	0.6	27.6	5.9	14.3	30.1	29.9	22.9	15.1
dc9	47.1	42.5	60.5	48.7	45.4	36.9	36.1	27.6	0.6	21.7	34.3	40.5	38.2	22.4	13.9
dc10	28	22.3	38.9	34.9	22.2	20.3	18.8	5.9	21.7	0.6	21	32.1	37.8	22.1	11.5
dc11	18.6	14.5	28.8	15.6	38.4	4.6	34.9	14.3	34.3	21	0.6	11.8	15.1	15.8	18.6
dc12	41.9	26.4	34.4	9.6	27.4	16.4	46.7	30.1	40.5	32.1	11.8	0.6	7.8	22.6	29.8
dc13	45	29.4	39.4	17.6	34.4	19.5	49.3	29.9	38.2	37.8	15.1	7.6	0.6	16.6	24.7
dc14	46.4	31.4	44.7	31.5	50.8	20.2	42.8	22.9	22.4	22.1	15.8	22.6	16.6	0.6	8.6
dc15	38.3	30.1	47.1	35.1	34.4	23	30.7	15.1	13.9	11.5	18.6	29.8	24.7	8.6	0.6
dc16	56.4	49.2	67.5	53.8	73.7	42.6	45.4	34.1	13.9	28.5	38.7	43.8	36.2	21.9	17.7

Table 26: dcrm-dt-co2

	rm1	rm2	rm3	rm4	rm5	rm6	rm7	rm8	rm9	rm10	rm11	rm12	rm13	rm14	rm15
dc1	9	257	353	602	344	397	299	323	721	429	459	643	690	705	586
dc2	257	9	276	344	89	151	386	250	651	342	223	405	451	4176	462
dc3	353	276	9	411	346	374	640	514	927	596	441	528	604	684	721
dc4	602	344	411	9	300	221	707	428	747	534	240	147	269	483	538
dc5	344	89	346	300	9	87	479	250	696	341	589	419	528	778	527
dc6	397	151	374	221	87	9	474	220	565	311	70	252	299	310	353
dc7	299	386	640	707	479	474	9	299	553	289	535	715	756	655	471
dc8	323	251	514	428	250	220	299	9	423	91	219	462	458	350	231
dc9	721	651	927	747	696	565	553	423	9	332	526	621	585	344	212
dc10	429	342	596	534	341	311	289	91	332	9	322	492	579	339	176
dc11	285	223	441	240	589	70	535	219	526	322	9	1176	231	242	285
dc12	643	405	528	147	419	252	715	462	621	492	1176	9	120	346	456
dc13	690	451	604	269	528	299	756	458	585	579	231	117	9	255	379
dc14	711	4176	684	483	778	310	655	350	344	339	242	346	255	9	132
dc15	586	462	721	538	527	353	471	231	212	176	285	456	378	132	9
dc16	865	753	1034	825	1129	653	696	523	213	436	594	671	554	335	271

Table 27: epdc-blocktrain-co2

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	438	342	414	362	NA	292	424	326	85	211	182	286	257	107	128	NA
ep2	251	155	227	178	NA	107	239	140	NA	258	133	NA	130	184	221	326
ep3	290	194	266	NA	NA	146	278	178	NA	264	108	178	219	223	235	364

Table 28: epdc-blocktrain-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	1700	1487	1664	1637	8029	1387	1664	1533	708	1058	970	1305	1221	678	787	8029
ep2	872	659	836	809	8029	558	836	704	8029	943	529	8029	829	901	631	1305
ep3	1046	833	1010	8029	8029	732	1046	878	8029	809	566	809	941	902	930	1637

Table 29: epdc-blocktrain-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	161	130	153	136	NA	114	156	125	28	87	78	112	102	54	61	NA
ep2	81	50	73	57	NA	35	77	45	NA	83	43	NA	42	59	71	124
ep3	93	62	86	NA	NA	47	89	57	NA	85	35	57	71	72	76	137

Table 30: epdc-hd1-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	2001	1774	2346	1920	2538	1571	1657	1306	676	1129	1450	1607	1369	923	793	100
ep2	1549	1066	1315	541	1094	755	1697	1181	1505	1242	609	100	486	947	1169	1607
ep3	1467	943	1079	100	852	691	1680	1114	1761	1329	730	541	789	1225	1336	1920

Table 31: epdc-hd1-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	293.1	255.4	350.7	279.7	382.7	221.4	235.8	177.3	72.3	147.8	201.3	227.5	187.8	113.6	91.8	3.2
ep2	217.9	137.3	178.9	49.9	142.1	85.4	242.6	156.5	210.6	166.7	61.1	3.2	40.6	117.4	154.6	227.5
ep3	204.2	116.8	139.5	3.2	101.8	74.9	239.7	145.3	253.1	181.1	81.3	49.9	91.2	163.8	182.4	279.7

Table 32: epdc-hd2-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	2167	1922	2542	2080	2750	1701	1795	1414	732	1223	1570	1741	1483	1000	859	108
ep2	1678	1154	1425	586	1186	817	1839	1279	1631	1346	659	108	526	1025	1267	1741
ep3	1589	1021	1169	108	923	749	1820	1206	1907	1439	790	586	855	1327	1448	2080

Table 33: epdc-hd2-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	58.6	51.1	70.1	55.9	76.5	44.3	47.2	35.5	14.5	29.6	40.3	45.5	37.6	22.7	18.4	0.6
ep2	43.6	27.5	35.8	10	28.4	17.1	48.5	31.3	42.1	33.3	12.2	0.6	8.1	23.5	30.9	45.5
ep3	40.8	23.4	27.9	0.6	20.4	15	47.9	29.1	50.6	36.2	16.3	10	18.2	32.8	36.5	55.9

Table 34: epdc-hd3-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	2417	2144	2835	2320	3067	1898	2002	1577	817	1364	1751	1942	1654	1116	958	121
ep2	1872	1288	1589	654	1322	912	2051	1427	1819	1501	735	121	587	1144	1413	1942
ep3	1772	1139	1304	121	1030	835	2030	1346	2127	1605	882	654	953	1480	1615	2320

Table 35: epdc-hd3-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	58.6	51.1	70.1	55.9	76.5	44.3	47.2	35.5	14.5	29.6	40.3	45.5	37.6	22.7	18.4	0.6
ep2	43.6	27.5	35.8	10	28.4	17.1	48.5	31.3	42.1	33.3	12.2	0.6	8.1	23.5	30.9	45.5
ep3	40.8	23.4	27.9	0.6	20.4	15	47.9	29.1	50.6	36.2	16.3	10	18.2	32.8	36.5	55.9

Table 36: epdc-hd4-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	3168	2809	3715	3040	4019	2487	2623	2067	1070	1787	2295	2544	2167	1462	1255	158
ep2	2453	1687	2082	857	1733	1195	2687	1869	2383	1967	964	158	769	1499	1851	2544
ep3	2322	1493	1708	158	1350	1094	2660	1763	2788	2104	1155	857	1249	1939	2116	3040

Table 37: epdc-hd4-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	29.3	25.5	35.1	28	38.3	22.1	23.6	17.7	7.2	14.8	20.1	22.8	18.8	11.4	9.2	0.3
ep2	21.8	13.7	17.9	5	14.2	8.5	24.3	15.6	21.1	16.7	6.1	0.3	4.1	11.7	15.5	22.8
ep3	20.4	11.7	14	0.3	10.2	7.5	24	14.5	25.3	18.1	8.1	5	9.1	16.4	18.2	28

Table 38: epdc-hd-co2

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	776	676	928	740	1013	586	624	469	191	391	533	602	497	301	243	8
ep2	577	363	473	132	376	226	642	414	557	441	162	8	108	311	409	602
ep3	540	309	369	8	269	198	634	385	670	479	215	132	241	434	483	740

Table 39: epdc-train-co2

	dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
ep1	438	342	414	362	NA	292	424	326	85	211	182	286	257	107	128	NA
ep2	251	155	227	178	NA	107	239	140	NA	258	133	NA	130	184	221	326
ep3	290	194	266	NA	NA	146	278	178	NA	264	108	178	219	223	235	364

Table 40: epdc-train-cost

	dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
ep1	2087	1822	2042	2009	NA	1696	2042	1878	847	1285	1175	1594	1488	810	946	NA
ep2	1052	787	1007	974	NA	660	1007	843	NA	1142	624	NA	998	1088	752	1594
ep3	1269	1004	1224	NA	NA	878	1269	1061	NA	974	671	974	1139	1090	1124	2009

Table 41: epdc-train-pm

	dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
ep1	161	130	153	136	NA	114	156	125	28	87	78	112	102	54	61	NA
ep2	81	50	73	57	NA	35	77	45	NA	83	43	NA	42	59	71	124
ep3	93	62	86	NA	NA	47	89	57	NA	85	35	57	71	72	76	137

Option B

Table 42: dcrm-dt1-cost

	rm1	rm2	rm3	rm4	rm5	rm6	rm7	rm8	rm9	rm10	rm11	rm12	rm13	rm14	rm15
dc1	176	1983	2143	2555	2127	2216	2054	2093	2752	2269	2318	2622	2701	2726	2529
dc2	1983	176	2015	2129	1705	1808	2197	1972	2636	2124	1927	2229	2305	2354	2322
dc3	2143	2015	176	2240	2132	2177	2618	2410	3093	2546	2288	2432	2558	2691	2752
dc4	2555	2129	2240	176	2055	1924	2729	2268	2794	2443	1955	1802	2004	2358	2449
dc5	2127	1705	2132	2055	176	1702	2352	1972	2710	2122	2533	2252	2432	2846	2430
dc6	2216	1808	2177	1924	1702	176	2343	1922	2494	2072	1674	1976	2054	2071	2143
dc7	2054	2197	2618	2729	2352	2343	176	2054	2474	2036	2444	2743	2810	2643	2338
dc8	2093	1974	2410	2268	1972	1922	2054	176	2258	1708	1921	2322	2316	2138	1941
dc9	2752	2636	3093	2794	2710	2494	2474	2258	176	2108	2429	2586	2527	2127	1910
dc10	2269	2124	2546	2443	2122	2072	2036	1708	2108	176	2091	2372	2516	2119	1851
dc11	2030	1927	2288	1955	2533	1674	2444	1921	2429	2091	176	1857	1941	1958	2030
dc12	2622	2229	2432	1802	2252	1976	2743	2322	2586	2372	1857	176	1757	2132	2313
dc13	2701	2305	2558	2004	2432	2054	2810	2316	2527	2516	1941	1752	176	1980	2185
dc14	2735	2354	2691	2358	2846	2071	2643	2138	2127	2119	1958	2132	1980	176	1777
dc15	2529	2322	2752	2449	2430	2143	2338	1941	1910	1851	2030	2313	2185	1777	176
dc16	2990	2805	3271	2924	3427	2640	2710	2424	1912	2280	2541	2669	2476	2113	2007

Table 43: dcrm-dt1-pm

	rm1	rm2	rm3	rm4	rm5	rm6	rm7	rm8	rm9	rm10	rm11	rm12	rm13	rm14	rm15
dc1	9.9	268.1	368.6	628.8	358.8	414.9	312.4	337.1	753	448.4	479	671.2	720.5	736.2	612.1
dc2	268.1	9.9	287.8	359.7	92.6	157.7	403.1	261.2	6176.1	356.8	232.6	422.8	471.1	501.7	482
dc3	368.6	287.8	9.9	429.7	361.7	390.3	668.2	537.2	967.9	622.9	460.3	551	630.8	714.6	753
dc4	628.8	359.7	429.7	9.9	313.4	230.6	738.2	447.5	779.6	557.8	250.3	153.8	2176.9	504.6	561.8
dc5	358.8	92.6	361.7	313.4	9.9	90.7	500.7	261.2	726.4	355.8	615	437.6	551	812.1	550
dc6	414.9	157.7	390.3	230.6	90.7	9.9	494.8	229.6	590.4	324.3	72.9	263.2	312.4	323.3	368.6
dc7	312.4	403.1	668.2	738.2	500.7	494.8	9.9	312.4	577.6	301.6	558.8	747.1	789.5	684	491.8
dc8	337.1	262.2	537.2	447.5	261.2	229.6	312.4	9.9	441.5	94.6	228.7	482	478	365.7	241.5
dc9	753	6176.1	967.9	779.6	726.4	590.4	577.6	441.5	9.9	346.9	549	648.5	611.1	358.8	221.8
dc10	448.4	356.8	622.9	557.8	355.8	324.3	301.6	94.6	346.9	9.9	336.1	513.5	604.2	353.8	184.3
dc11	297.7	232.6	460.3	250.3	615	72.9	558.8	228.7	549	336.1	9.9	188.2	241.5	252.3	297.7
dc12	671.2	422.8	551	153.8	437.6	263.2	747.1	482	648.5	513.5	188.2	9.9	125.2	361.7	476
dc13	720.5	471.1	630.8	2176.9	551	312.4	789.5	478	611.1	604.2	241.5	122.2	9.9	266.1	395.3
dc14	742.2	501.7	714.6	504.6	812.1	323.3	684	365.7	358.8	353.8	252.3	361.7	266.1	9.9	138
dc15	612.1	482	753	561.8	550	368.6	491.8	241.5	221.8	184.3	297.7	476	395.2	138	9.9
dc16	903	786.5	1080.2	861.4	1178.8	682	726.4	546	222.7	455.3	619.9	700.8	578.5	349.9	282.9

Table 44: dcrm-dt2-cost

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	191	2148	2321	2767	2304	2400	2224	2266	2980	2458	2510	2840	2925	2952	2739
dc2	2148	191	2182	2305	1847	1958	2380	2136	2855	2300	2087	2414	2497	2549	2515
dc3	2321	2182	191	2425	2309	2358	2835	2610	3349	2757	2478	2634	2771	2915	2980
dc4	2767	2305	2425	191	2226	2084	2955	2456	3026	2645	2118	1952	2170	2554	2652
dc5	2304	1847	2309	2226	191	1843	2547	2136	2935	2299	2744	2439	2634	3082	2632
dc6	2400	1958	2358	2084	1843	191	2537	2082	2701	2244	1813	2140	2224	2243	2321
dc7	2224	2380	2835	2955	2547	2537	191	2224	2679	2206	2647	2970	3043	2862	2532
dc8	2266	2138	2610	2456	2136	2082	2224	191	2446	1850	2080	2515	2508	2316	2102
dc9	2980	2855	3349	3026	2935	2701	2679	2446	191	2283	2630	2801	2737	2304	2068
dc10	2458	2300	2757	2645	2299	2244	2206	1850	2283	191	2265	2569	2725	2295	2004
dc11	2199	2087	2478	2118	2744	1813	2647	2080	2630	2265	191	2011	2102	2121	2199
dc12	2840	2414	2634	1952	2439	2140	2970	2515	2801	2569	2011	191	1903	2309	2505
dc13	2925	2497	2771	2170	2634	2224	3043	2508	2737	2725	2102	1898	191	2145	2366
dc14	2962	2549	2915	2554	3082	2243	2862	2316	2304	2295	2121	2309	2145	191	1925
dc15	2739	2515	2980	2652	2632	2321	2532	2102	2068	2004	2199	2505	2366	1925	191
dc16	3238	3038	3542	3167	3711	2859	2935	2625	2070	2469	2752	2891	2681	2288	2173

Table 45: dcrm-dt2-pm

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	2	53.6	73.7	125.8	71.8	83	62.5	67.4	150.6	89.7	95.8	134.2	144.1	147.2	122.4
dc2	53.6	2	57.6	71.9	18.5	31.5	176.6	52.2	136	71.4	46.5	84.6	94.2	100.3	96.4
dc3	73.7	57.6	2	85.9	72.3	78.1	133.6	107.4	193.6	124.6	92.1	110.2	126.2	142.9	150.6
dc4	125.8	71.9	85.9	2	62.7	46.1	147.6	89.5	155.9	111.6	50.1	30.8	56.2	100.9	112.4
dc5	71.8	18.5	72.3	62.7	2	18.1	100.1	52.2	145.3	71.2	123	87.5	110.2	162.4	110
dc6	83	31.5	78.1	46.1	18.1	2	99	45.9	118.1	64.9	14.6	52.6	62.5	64.7	73.7
dc7	62.5	176.6	133.6	147.6	100.1	99	2	62.5	115.5	60.3	111.8	149.4	157.9	136.8	98.4
dc8	67.4	52.4	107.4	89.5	52.2	45.9	62.5	2	88.3	18.9	45.7	96.4	95.6	73.1	48.3
dc9	150.6	136	193.6	155.9	145.3	118.1	115.5	88.3	2	69.4	109.8	129.7	122.2	71.8	44.4
dc10	89.7	71.4	124.6	111.6	71.2	64.9	60.3	18.9	69.4	2	67.2	102.7	120.8	70.8	36.9
dc11	59.5	46.5	92.1	50.1	123	14.6	111.8	45.7	109.8	67.2	2	37.6	48.3	50.5	59.5
dc12	134.2	84.6	110.2	30.8	87.5	52.6	149.4	96.4	129.7	102.7	37.6	2	25	72.3	95.2
dc13	144.1	94.2	126.2	56.2	110.2	62.5	157.9	95.6	122.2	120.8	48.3	24.4	2	53.2	79.1
dc14	148.4	100.3	142.9	100.9	162.4	64.7	136.8	73.1	71.8	70.8	50.5	72.3	53.2	2	27.6
dc15	122.4	96.4	150.6	112.4	110	73.7	98.4	48.3	44.4	36.9	59.5	95.2	79	27.6	2
dc16	1176.6	157.3	216	172.3	235.8	136.4	145.3	109.2	44.5	91.1	124	140.2	115.7	70	56.6

Table 46: dcrm-dt3-cost

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	206	2317	2503	2984	2484	2588	2399	2444	3214	2651	2707	3063	3154	3183	2953
dc2	2317	206	2353	2486	1992	2112	2567	2304	3079	2481	2251	2603	2693	2749	2713
dc3	2503	2353	206	2616	2490	2543	3057	2815	3612	2974	2672	2840	2988	3143	3214
dc4	2984	2486	2616	206	2401	2247	3187	2649	3264	2853	2284	2105	2340	2755	2860
dc5	2484	1992	2490	2401	206	1988	2747	2304	3165	2479	2959	2630	2840	3324	2838
dc6	2588	2112	2543	2247	1988	206	2736	2245	2913	2421	1955	2307	2399	2419	2503
dc7	2399	2567	3057	3187	2747	2736	206	2399	2890	2379	2855	3203	3282	3087	2731
dc8	2444	2306	2815	2649	2304	2245	2399	206	2638	1995	2244	2713	2705	2497	2267
dc9	3214	3079	3612	3264	3165	2913	2890	2638	206	2463	2837	3021	2952	2484	2231
dc10	2651	2481	2974	2853	2479	2421	2379	1995	2463	206	2442	2771	2939	2475	2161
dc11	2371	2251	2672	2284	2959	1955	2855	2244	2837	2442	206	2169	2267	2287	2371
dc12	3063	2603	2840	2105	2630	2307	3203	2713	3021	2771	2169	206	2052	2490	2702
dc13	3154	2693	2988	2340	2840	2399	3282	2705	2952	2939	2267	2046	206	2313	2552
dc14	3194	2749	3143	2755	3324	2419	3087	2497	2484	2475	2287	2490	2313	206	2076
dc15	2953	2713	3214	2860	2838	2503	2731	2267	2231	2161	2371	2702	2552	2076	206
dc16	3492	3276	3820	3415	4003	3083	3165	2831	2233	2663	2968	3118	2891	2468	2344

Table 47: dcrm-dt3-pm

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	2	53.6	73.7	125.8	71.8	83	62.5	67.4	150.6	89.7	95.8	134.2	144.1	147.2	122.4
dc2	53.6	2	57.6	71.9	18.5	31.5	176.6	52.2	136	71.4	46.5	84.6	94.2	100.3	96.4
dc3	73.7	57.6	2	85.9	72.3	78.1	133.6	107.4	193.6	124.6	92.1	110.2	126.2	142.9	150.6
dc4	125.8	71.9	85.9	2	62.7	46.1	147.6	89.5	155.9	111.6	50.1	30.8	56.2	100.9	112.4
dc5	71.8	18.5	72.3	62.7	2	18.1	100.1	52.2	145.3	71.2	123	87.5	110.2	162.4	110
dc6	83	31.5	78.1	46.1	18.1	2	99	45.9	118.1	64.9	14.6	52.6	62.5	64.7	73.7
dc7	62.5	176.6	133.6	147.6	100.1	99	2	62.5	115.5	60.3	111.8	149.4	157.9	136.8	98.4
dc8	67.4	52.4	107.4	89.5	52.2	45.9	62.5	2	88.3	18.9	45.7	96.4	95.6	73.1	48.3
dc9	150.6	136	193.6	155.9	145.3	118.1	115.5	88.3	2	69.4	109.8	129.7	122.2	71.8	44.4
dc10	89.7	71.4	124.6	111.6	71.2	64.9	60.3	18.9	69.4	2	67.2	102.7	120.8	70.8	36.9
dc11	59.5	46.5	92.1	50.1	123	14.6	111.8	45.7	109.8	67.2	2	37.6	48.3	50.5	59.5
dc12	134.2	84.6	110.2	30.8	87.5	52.6	149.4	96.4	129.7	102.7	37.6	2	25	72.3	95.2
dc13	144.1	94.2	126.2	56.2	110.2	62.5	157.9	95.6	122.2	120.8	48.3	24.4	2	53.2	79.1
dc14	148.4	100.3	142.9	100.9	162.4	64.7	136.8	73.1	71.8	70.8	50.5	72.3	53.2	2	27.6
dc15	122.4	96.4	150.6	112.4	110	73.7	98.4	48.3	44.4	36.9	59.5	95.2	79	27.6	2
dc16	1176.6	157.3	216	172.3	235.8	136.4	145.3	109.2	44.5	91.1	124	140.2	115.7	70	56.6

Table 48: dcrm-dt4-cost

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	308	3471	3750	4472	3722	3878	3594	3662	4816	3971	4056	4589	4726	4770	4425
dc2	3471	308	3526	3725	2984	3165	3845	3452	4614	3717	3372	3900	4034	4119	4064
dc3	3750	3526	308	3919	3731	3810	4581	4217	5412	4455	4004	4256	4477	4709	4816
dc4	4472	3725	3919	308	3597	3367	4775	3969	4890	4275	3422	3154	3506	4127	4286
dc5	3722	2984	3731	3597	308	2979	4116	3452	4742	3714	4433	3941	4256	4980	4253
dc6	3878	3165	3810	3367	2979	308	4100	3364	4365	3627	2930	3457	3594	3624	3750
dc7	3594	3845	4581	4775	4116	4100	308	3594	4329	3564	4277	4800	4917	4625	4092
dc8	3662	3454	4217	3969	3452	3364	3594	308	3952	2990	3362	4064	4053	3742	3397
dc9	4816	4614	5412	4890	4742	4365	4329	3952	308	3690	4250	4526	4422	3722	3342
dc10	3971	3717	4455	4275	3714	3627	3564	2990	3690	308	3660	4152	4403	3709	3238
dc11	3553	3372	4004	3422	4433	2930	4277	3362	4250	3660	308	3249	3397	3427	3553
dc12	4589	3900	4256	3154	3941	3457	4800	4064	4526	4152	3249	308	3074	3731	4048
dc13	4726	4034	4477	3506	4256	3594	4917	4053	4422	4403	3397	3066	308	3465	3824
dc14	4786	4119	4709	4127	4980	3624	4625	3742	3722	3709	3427	3731	3465	308	3110
dc15	4425	4064	4816	4286	4253	3750	4092	3397	3342	3238	3553	4048	3824	3110	308
dc16	5232	4909	5724	5117	5997	4619	4742	4242	3345	3990	4447	4671	4332	3698	3512

Table 49: dcrm-dt4-pm

	rm1	**rm2**	**rm3**	**rm4**	**rm5**	**rm6**	**rm7**	**rm8**	**rm9**	**rm10**	**rm11**	**rm12**	**rm13**	**rm14**	**rm15**
dc1	1	26.8	36.9	62.9	35.9	41.5	31.2	33.7	75.3	44.8	47.9	67.1	72	73.6	61.2
dc2	26.8	1	28.8	36	9.3	15.8	40.3	26.1	68	35.7	23.3	42.3	47.1	50.2	48.2
dc3	36.9	28.8	1	43	36.2	39	66.8	53.7	96.8	62.3	46	55.1	63.1	71.5	75.3
dc4	62.9	36	43	1	31.3	23.1	73.8	44.7	78	55.8	25	15.4	28.1	50.5	56.2
dc5	35.9	9.3	36.2	31.3	1	9.1	50.1	26.1	72.6	35.6	61.5	43.8	55.1	81.2	55
dc6	41.5	15.8	39	23.1	9.1	1	49.5	23	59	32.4	7.3	26.3	31.2	32.3	36.9
dc7	31.2	40.3	66.8	73.8	50.1	49.5	1	31.2	57.8	30.2	55.9	74.7	78.9	68.4	49.2
dc8	33.7	26.2	53.7	44.7	26.1	23	31.2	1	44.2	9.5	22.9	48.2	47.8	36.6	24.1
dc9	75.3	68	96.8	78	72.6	59	57.8	44.2	1	34.7	54.9	64.9	61.1	35.9	22.2
dc10	44.8	35.7	62.3	55.8	35.6	32.4	30.2	9.5	34.7	1	33.6	51.3	60.4	35.4	18.4
dc11	29.8	23.3	46	25	61.5	7.3	55.9	22.9	54.9	33.6	1	18.8	24.1	25.2	29.8
dc12	67.1	42.3	55.1	15.4	43.8	26.3	74.7	48.2	64.9	51.3	18.8	1	12.5	36.2	47.6
dc13	72	47.1	63.1	28.1	55.1	31.2	78.9	47.8	61.1	60.4	24.1	12.2	1	26.6	39.5
dc14	74.2	50.2	71.5	50.5	81.2	32.3	68.4	36.6	35.9	35.4	25.2	36.2	26.6	1	13.8
dc15	61.2	48.2	75.3	56.2	55	36.9	49.2	24.1	22.2	18.4	29.8	47.6	39.5	13.8	1
dc16	90.3	78.7	108	86.1	117.9	68.2	72.6	54.6	22.3	45.5	62	70.1	57.9	35	28.3

Table 50: dcrm-dt-co2

	rm1	rm2	rm3	rm4	rm5	rm6	rm7	rm8	rm9	rm10	rm11	rm12	rm13	rm14	rm15
dc1	15	411	565	963	550	636	479	516	1154	687	734	1028	1104	1128	938
dc2	411	15	441	551	142	242	618	400	1042	547	356	648	722	769	738
dc3	565	441	15	658	554	598	1024	823	1483	954	705	844	966	1095	1154
dc4	963	551	658	15	480	353	1131	686	1194	855	384	236	430	773	861
dc5	550	142	554	4176	15	139	767	400	1113	545	942	670	844	1244	843
dc6	636	242	598	353	139	15	758	352	905	497	112	403	479	495	565
dc7	479	618	1024	1131	767	758	15	479	885	462	856	1145	1210	1048	754
dc8	516	402	823	686	400	352	479	15	677	145	350	738	732	560	370
dc9	1154	1042	1483	1194	1113	905	885	677	15	532	841	994	936	550	340
dc10	687	547	954	855	545	497	462	145	532	15	515	787	926	542	282
dc11	456	356	705	384	942	112	856	350	841	515	15	288	370	387	456
dc12	1028	648	844	236	670	403	1145	738	994	787	288	15	192	554	729
dc13	1104	722	966	430	844	479	1210	732	936	926	370	187	15	408	606
dc14	1137	769	1095	773	1244	495	1048	560	550	542	387	554	408	15	211
dc15	938	738	1154	861	843	565	754	370	340	282	456	729	606	211	15
dc16	1383	1205	1655	1320	1806	1045	1113	837	341	698	950	1074	886	536	433

Table 51: epdc-blocktrain-co2

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	438	342	414	362	NA	292	424	326	85	211	182	286	257	107	128	NA
ep2	251	155	227	178	NA	107	239	140	NA	258	133	NA	130	184	221	326
ep3	290	194	266	NA	NA	146	278	178	NA	264	108	178	219	223	235	364

Table 52: epdc-blocktrain-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	1700	1487	1664	1637	8029	1387	1664	1533	708	1058	970	1305	1221	678	787	8029
ep2	872	659	836	809	8029	558	836	704	8029	943	529	8029	829	901	631	1305
ep3	1046	833	1010	8029	8029	732	1046	878	8029	809	566	809	941	902	930	1637

Table 53: epdc-blocktrain-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	161	130	153	136	NA	114	156	125	28	87	78	112	102	54	61	NA
ep2	81	50	73	57	NA	35	77	45	NA	83	43	NA	42	59	71	124
ep3	93	62	86	NA	NA	47	89	57	NA	85	35	57	71	72	76	137

Table 54: epdc-hd1-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	1600	1419	1877	1536	2031	1256	1326	1044	541	903	1160	1286	1095	739	634	80
ep2	1240	852	1052	433	875	604	1358	945	1204	994	487	80	389	757	935	1286
ep3	1173	754	863	80	682	553	1344	891	1408	1063	584	433	631	980	1069	1536

Table 55: epdc-hd1-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	498.3	434.1	596.2	475.5	650.6	376.4	400.9	301.4	122.9	251.3	342.2	386.8	319.3	193.1	156.1	5.4
ep2	370.5	233.4	304.1	84.9	241.5	145.2	412.4	266	358	283.4	103.9	5.4	69.1	199.6	262.8	386.8
ep3	347.1	198.6	237.2	5.4	173	127.3	407.5	247	430.3	307.9	138.2	84.9	155	278.5	310.1	475.5

Table 56: epdc-hd2-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	1734	1537	2033	1664	2200	1361	1436	1131	586	978	1256	1393	1186	800	687	87
ep2	1343	923	1140	469	948	654	1471	1023	1304	1077	527	87	421	820	1013	1393
ep3	1271	817	935	87	739	599	1456	965	1526	1151	632	469	684	1062	1158	1664

Table 57: epdc-hd2-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	99.7	86.8	119.2	95.1	130.1	75.3	176.2	60.3	24.6	50.3	68.4	77.4	63.9	38.6	31.2	1.1
ep2	74.1	46.7	60.8	17	48.3	29	82.5	53.2	71.6	56.7	20.8	1.1	13.8	39.9	52.6	77.4
ep3	69.4	39.7	47.4	1.1	34.6	25.5	81.5	49.4	86.1	61.6	27.6	17	31	55.7	62	95.1

Table 58: epdc-hd3-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	1934	1715	2268	1856	2454	1518	1602	1262	653	1091	1401	1553	1323	893	766	97
ep2	1498	1030	1271	523	1058	729	1641	1141	1455	1201	588	97	470	915	1130	1553
ep3	1418	911	1043	97	824	668	1624	1076	1702	1284	705	523	763	1184	1292	1856

Table 59: epdc-hd3-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	99.7	86.8	119.2	95.1	130.1	75.3	176.2	60.3	24.6	50.3	68.4	77.4	63.9	38.6	31.2	1.1
ep2	74.1	46.7	60.8	17	48.3	29	82.5	53.2	71.6	56.7	20.8	1.1	13.8	39.9	52.6	77.4
ep3	69.4	39.7	47.4	1.1	34.6	25.5	81.5	49.4	86.1	61.6	27.6	17	31	55.7	62	95.1

Table 60: epdc-hd4-cost

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	2534	2247	2972	2432	3215	1989	2099	1654	856	1430	1836	2035	1734	1170	1004	127
ep2	1963	1350	1666	686	1386	956	2150	1496	1907	1573	771	127	615	1199	1481	2035
ep3	1858	1194	1367	127	1080	875	2128	1410	2230	1683	924	686	999	1552	1693	2432

Table 61: epdc-hd4-pm

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	49.8	43.4	59.6	47.5	65.1	37.6	40.1	30.1	12.3	25.1	34.2	38.7	31.9	19.3	15.6	0.5
ep2	37	23.3	30.4	8.5	24.2	14.5	41.2	26.6	35.8	28.3	10.4	0.5	6.9	20	26.3	38.7
ep3	34.7	19.9	23.7	0.5	17.3	12.7	40.7	24.7	43	30.8	13.8	8.5	15.5	27.9	31	47.5

Table 62: epdc-hd-co2

	dc1	dc2	dc3	dc4	dc5	dc6	dc7	dc8	dc9	dc10	dc11	dc12	dc13	dc14	dc15	dc16
ep1	1319	1149	1578	1258	1722	996	1061	798	325	665	906	1024	845	511	413	14
ep2	981	618	805	225	639	384	1091	704	947	750	275	14	183	528	695	1024
ep3	919	526	628	14	458	337	1078	654	1139	815	366	225	410	737	821	1258

Table 63: epdc-train-co2

	dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
ep1	438	342	414	362	NA	292	424	326	85	211	182	286	257	107	128	NA
ep2	251	155	227	178	NA	107	239	140	NA	258	133	NA	130	184	221	326
ep3	290	194	266	NA	NA	146	278	178	NA	264	108	178	219	223	235	364

Table 64: epdc-train-cost

	dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
ep1	2087	1822	2042	2009	NA	1696	2042	1878	847	1285	1175	1594	1488	810	946	NA
ep2	1052	787	1007	974	NA	660	1007	843	NA	1142	624	NA	998	1088	752	1594
ep3	1269	1004	1224	NA	NA	878	1269	1061	NA	974	671	974	1139	1090	1124	2009

Table 65: epdc-train-pm

	dc1	**dc2**	**dc3**	**dc4**	**dc5**	**dc6**	**dc7**	**dc8**	**dc9**	**dc10**	**dc11**	**dc12**	**dc13**	**dc14**	**dc15**	**dc16**
ep1	161	130	153	136	NA	114	156	125	28	87	78	112	102	54	61	NA
ep2	81	50	73	57	NA	35	77	45	NA	83	43	NA	42	59	71	124
ep3	93	62	86	NA	NA	47	89	57	NA	85	35	57	71	72	76	137