Facing Robustness as a Multi-objective Problem: A Bi-objective Shortest Path Problem in Smart Regions

Article *in* Information Sciences · July 2019

DOI: 10.1016/j.ins.2019.07.014

CITATIONS
11

READS
320

3 authors:

Christian Cintrano
University of Malaga
21 PUBLICATIONS **77** CITATIONS
SEE PROFILE

Francisco Chicano
University of Malaga
174 PUBLICATIONS **2,498** CITATIONS
SEE PROFILE

Enrique Alba
University of Malaga
643 PUBLICATIONS **17,823** CITATIONS
SEE PROFILE

Facing Robustness as a Multi-objective Problem: A Bi-objective Shortest Path Problem in Smart Regions

C. Cintrano[a,*], F. Chicano[a], E. Alba[a]

[a] *Universidad de Málaga, E.T.S.I. Informática, Bulevar Louis Pasteur 35, 29071 Malaga, Spain*

Abstract

The goal in Robust Optimization is to optimize not only the quality of the solutions but also the variation of this quality with the uncertain parameters of the optimization problem. We propose a robust model for the bi-objective shortest path problem applied in a smart mobility context: finding routes for cars in a city to minimize travel time and gas emissions. Our proposal treats robustness from a multi-objective point of view. We model the parameters that define each instance as random variables, described through their mean and variance. In this way, we can obtain efficient solutions that are also less sensitive to changes in the environment. We run different types of algorithms in multiple instances to solve this problem so that we obtain a global view of the behavior of different techniques. All experimentation uses a scenario based on real data: the province of Malaga, Spain. This realistic settlement for our study allows us to test the applicability of our model in final systems for the citizens.

The results clearly state the interest of our proposal for tackling robustness and represents a new state-of-the-art in smart mobility, an always appealing feature of works, that could lead to an industrial prototype.

Keywords: Robustness, traffic road network, bi-objective shortest path, multi-objective optimization

1. Introduction

Millions of citizens around the globe have to face every day with the problem of deciding the route to follow in their cities to go from an origin (e.g., home) to a destination (e.g., work). Many software tools solve this problem by representing the city as a graph and assigning a cost to every edge (street segment) in the graph. Then, Dijkstra shortest path (SP) algorithm [9] is used to find the route minimizing the cost function. The time to traverse the route is commonly used as cost. This is what navigator tools like Tomtom, Google Maps, or Waze do. Distance is also a common parameter that can be easily minimized using this approach.

However, in real world, reducing the travel time (or the distance) is not the only important goal. Many citizens also want to spend less money on fuel, drive through quiet streets, and, in general, fulfill several of these criteria at the same time. A shortest path problem that simultaneously deals with two objectives is called a bi-objective shortest path (BSP) problem. Sometimes the data required to solve the problem is not reliable or accurate. For example, the time needed to traverse a street depends on the traffic load, the vehicles used, the speed limit, the state of the traffic lights, and even the skills of the driver. Often, all of these parameters are not accurately known. For example, if a citizen goes to work, s/he normally prefers the fastest route, however, the main roads usually have traffic jams in rush hour, so the time may be longer than another route with higher distance but with less traffic. It is important that the solutions proposed for this kind of scenarios consider the uncertainty in the data. That is, given a solution, we would like to know how much its quality attributes can change due to the variation of the uncertain data we have about the city. This quality of the solver technique and problem definition is usually called *robustness* [2].

In this work, we include robustness in our new model of the BSP problem to take into account the uncertainty in the data. The objectives we consider are: travel time and the gas emission (CO_2 and NO_x). We select these two goals guided by our present societies and modern needs. We need to waste as little time as possible on travelling and pollution problems in cities affect the health of their citizens. Our contributions in this work can be summarized as follows:

*Corresponding author

Email addresses: cintrano@lcc.uma.es (C. Cintrano), chicano@lcc.uma.es (F. Chicano), eat@lcc.uma.es (E. Alba)

- We introduce a robustness model into the definition of the BSP itself. We model an optimization problem that searches for routes with the lowest average travel time and gas emitted, and also for the smallest variability of these two goals. This new formulation including robustness for this problem is not present in the literature, where usually one single robust solution is returned by the algorithms [4, 20, 31].
- We use explicit multi-objective modeling of the problem along with a shift from fixed parameters to random variables.
- We apply different algorithms to solve the multi-objective problem in order to check their performance. Multi-objective algorithms and weighted sums in single-objective algorithms are the chosen strategies.
- We use a real map with real data in our study. We select the Malaga Province, Spain, as the base map to our experiments. In addition to being a large scenario full of real data, it also represents a case study in a traditional European region, with interesting implications in creating services for tourists and commuters.

This work extends the previous conference paper [7], which was the first contact with the robust bi-objective shortest path problem, in which all values for the parameters, time and pollution, were randomly generated. We used three algorithms: Pulse [10], Dijkstra's algorithm, and A*, the last two without taking into account the robustness. In the present work, we extend that preliminary study to include two multi-objective algorithms: NSGA-II and MOEA/D. Unlike the previous work, all the algorithms used in this study solve the robust formulation of the problem. We also use twenty-nine different instances with real data, and an in-depth analysis of the different results obtained. In addition, this work studies the use of single-objective algorithms to solve multi-objective problems using weighed sums to get an approximation of the Pareto optimal set.

This paper is organized as follows. Section 2 defines some concepts of multi-objective optimization and mathematically formulates the BSP problem. Section 3 describes our proposed model to the robust BSP problem. Section 4 presents the information (data, algorithms, and computer platform) necessary to replicate our work and understand our results. Section 5 discusses the main results obtained by experimentation. Section 6 describes different aspects of robustness related to the present work. Finally, Section 7 presents the conclusions and some lines of future work.

2. Background

Our model treats the robustness of the problem as a variability in the parameters that must be minimized to obtain more robust routes. Therefore, applying multi-objective optimization techniques is a natural step towards solving the robust BSP problem. Before we can define our model of robustness, we must present some necessary mathematical definitions related to multi-objective optimization and the bi-objective shortest path problem to properly understand our proposal. Let us first define a basic optimization problem.

Definition 1 (Optimization problem). *An optimization problem consists in finding a solution $x \in X$ that minimizes an objective function $f : X \to \mathbb{R}^d$, where X is the search space.*

Each candidate solution generated by an algorithm can be evaluated using the function f. If the image of f is $\mathbb{R}$ ($d = 1$), we have a single-objective optimization problem. However, if $d > 1$ then we say that the problem is multi-objective. In the latter case, we will use boldface for the objective function $\mathbf{f}$ to highlight that it is a vector function. In our case, the fitness function of the non-robust model of the problem calculates the travel time and the gas emission produced by the vehicle ($d = 2$).

In a multi-objective problem, each objective is independent, equally important, and opposed to the other ones, i.e., we can get solutions with low travel time but with high gas emission, and vice versa. In this kind of problems, we can have several optimal solutions. *Pareto dominance* is a useful concept to define this set of optimal solutions. Formally, we define it as follows.

Definition 2 (Pareto dominance). *Given a vector function $\mathbf{f} : X \to \mathbb{R}^d$, we say that solution $x \in X$* dominates *solution $y \in X$, denoted with $x \prec_{\mathbf{f}} y$, if and only if $f_i(x) \leq f_i(y)$ for all $1 \leq i \leq d$ and there exists $j \in \{1, 2, \ldots, d\}$ such that $f_j(x) < f_j(y)$. When the vector function is clear from the context, we will use $\prec$ instead of $\prec_{\mathbf{f}}$.*

Pareto dominance defines a partial order relationship in the solution space. Thus, one route will be better than another if it has a lower travel time and gas emission than the other route.

Definition 3 (Pareto Optimal Set and Pareto Front). *Given a vector function* $\mathbf{f} : X \to \mathbb{R}^d$, *the* Pareto Optimal Set *is the set of solutions* $Y \subseteq X$ *that are not dominated by any other in* X*:*

$$Y = \{x \in X | \nexists y \in X,\ y \prec x\} \tag{1}$$

The Pareto Front *is the image by* $\mathbf{f}$ *of the Pareto Optimal Set:* $PF = \mathbf{f}(Y)$.

Definition 4 (Set of Non-dominated Solutions). *Given a vector function* $\mathbf{f} : X \to \mathbb{R}^d$, *we say that a set* $Y \subseteq X$ *is a* set of non-dominated solutions *if there is not a pair of solutions in the set in which one solution dominates another. In other words, there is no pair of solutions* $x, y \in Y$ *where* $y \prec x$, *i.e.,* $\forall x \in Y,\ \nexists y \in Y,\ y \prec x$.

If a solution x is dominated, it means that there is at least one solution y that it is better than x in each objective. For that reason, algorithms should try to find only non-dominated solutions.

Now, we have the necessary background to formally describe the problem solved in this work. Let $G(N, A)$ be a directed graph, where N is the set of nodes, and A the set of edges between nodes, $A \subseteq N \times N$. We define a path p with n nodes as a sequence of nodes $p_i \in N$ with $1 \leq i \leq n$, where $(p_i, p_{i+1}) \in A$ for $1 \leq i \leq n-1$. That is, consecutive nodes in the sequence are adjacent in the graph. We define $\mathcal{P}_{s,e}$ as the set of all possible paths between a start node $s = p_1$ and an end node $e = p_n$. In our problem, the graph represents the road map. The edges are the street segments between every two intersections (nodes of the graph). This is a classic way of modeling a road map [14], which allows us to formalize the real problem, while we keep its main features.

We also define a cost function $C : A \to \mathbb{R}^+$ in the graph G. This function assigns a non-negative number to each edge of the graph. In order to simplify the formulation, we write $C((i, j)) = c_{ij}$ as the cost of the edge (i, j). These costs represent the values of the objectives to be minimized, e.g., the values of travel time and emission in each of the road segments. The definition of the cost function can be extended to a whole path as follows:

$$z(p) = \sum_{(i,j) \in p} c_{ij}, \tag{2}$$

where we write $(i, j) \in p$ when nodes i and j are consecutive in path p.

In this work, we present a model of robustness for the Bi-objective Shortest Path (BSP) problem. BSP problem is an NP-hard multi-objective optimization problem [36] that searches for the paths between two points in a graph $G(N, A)$ minimizing two objectives simultaneously in the Pareto sense. As we have already explained, this graph problem is analogous to our routing problem. The cost function is defined as $C : A \to \mathbb{R}^+ \times \mathbb{R}^+$, which associates two weights to each edge in the graph. As in the single-objective case, we simplify the formulation by writing $C((i, j)) = c_{ij,k}$ to represent the cost of edge (i, j) in the objective k. We formulate the BSP problem as follows.

Definition 5 (Bi-objective Shortest Path Problem). *Given a graph* $G(N, A)$ *and two nodes in the graph* s *and* e, *the bi-objective shortest path problem consists in finding the Pareto optimal set of the bi-objective problem defined over the objective function:*

$$\mathbf{z}(p) = \sum_{(i,j) \in p} (c_{ij,1}, c_{ij,2}) \tag{3}$$

subject to $p \in \mathcal{P}_{s,e}$, *that is, only paths from nodes* s *and* e *are considered.*

Once we have introduced the necessary background for our proposal we can present our robust model for the BSP problem.

3. Proposed Robust Model

We call *robustness of a problem* to the characteristic of the problem to deal with inaccuracies in the parameters that defining an instance of the problem. The street length, travel times, or the map itself in a routing problem are examples of parameters of a problem. We say that a problem formulation is robust if it considers the inaccuracies in the parameters. Robustness in problem formulations is an important characteristic to take into account when we develop real-world applications. Our way to introduce robustness in the formulation is different from most other models because we consider robustness as additional objectives to be optimized.

In a general optimization problem, the fitness function f usually needs some additional information about the specific instance of the problem to be solved. These parameters of the problem

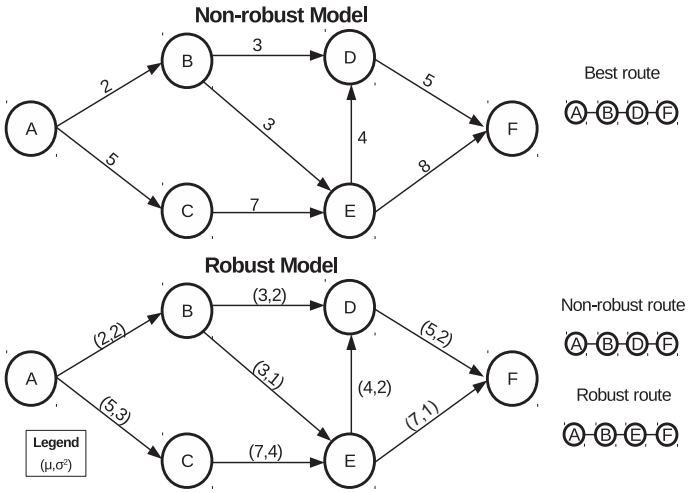


Figure 1: Standard non-robust model (top) and our proposed robust model (bottom). We also show the best route for the first one and two routes for the second.

allow us to instantiate our problem to a particular case. We can consider the parameters of the problem as another argument of the objective function. The new objective function has the form $f^P : X \times P \to \mathbb{R}^d$, where P is the space of parameters. P is a tuple $P = (P_1,\ P_2\ , \ldots,\ P_m)$, where P_i with $1 \leq i \leq m$ is the domain for the i-th parameter of the problem.

In real-world applications for the cities, we do not have the same control over the environment of the problem as in the lab. The real data usually have inaccuracies, missing information, etc. There are many strategies to deal with these inaccuracies (see Section 6). In this work, we assume that the parameters are random variables and we work with two statistics of them: their mean and variance. This is different from previous studies because it takes into account both, the general character of the road and their variation. Given the i-th parameter of the problem with domain $P_i \subseteq \mathbb{R}$, we consider that its value is defined by a random variable with mean μ_i and variance σ_i^2. We say that the parameter *inaccurate* if $\sigma_i^2 > 0$.

In a Shortest Path problem the cost function $C : A \to \mathbb{R}^+$ is a parameter of the problem that can have uncertainties. Thus, the cost associated to edge (i, j) is modelled with a random variable with mean μ_{ij} and variance σ_{ij}^2. We will assume that random variables associated with different edges are independent, what allows us to sum means and variances to get the mean and variance of the sum of costs. In Figure 1 we can see an example of this model of robustness. Given a graph $G(N, E)$, with $N = \{A, B, \ldots, F\}$, the shortest path between nodes A and F is calculated. In the top network, the shortest route (using the cost of each edge) has a cost of 10. In the bottom network, the variance associated with each edge is introduced (the robust version of the problem). Two routes are relevant in this case. The first is (A, B, D, F) with $\mu = 10$ and $\sigma^2 = 6$. The second one is (A, B, E, F) with $\mu = 12$ and $\sigma^2 = 4$. The first one has a lower mean cost but higher variance, while the second has a lower variance and higher mean cost. We say that the second solution is more robust than the first one since the variation of its cost due to the uncertainties in the parameters of the problem is lower than in the case of the first solution.

In the case of the BSP, we consider two parameters for each edge. These parameters are travel time (TT) and gas emission (CO_2 and NO_x). Each of them are characterized by a mean $\mu_{ij,k}$ and a variance $\sigma_{ij,k}^2$, with k being TT for travel time and GAS for gas emission. Our robust formulation for the bi-objective shortest path (RBSP) has a total of four objectives that are defined as follows:

$$\mathbf{z}^R(p) = \sum_{(i,j)\in p} (\mu_{ij,TT},\ \mu_{ij,GAS},\ \sigma_{ij,TT}^2,\ \sigma_{ij,GAS}^2), \tag{4}$$

where $p \in \mathcal{P}_{s,e}$ for a given s and e defined by the instance. In this way, we shift from a basic bi-objective problem to another one with four objectives. This formulation allows us to obtain solutions with different degrees of robustness (greater or lower variance) in both parameters (travel time or gas emission). Having one set of solutions is interesting for many types of applications. According to the circumstances, a user may need solutions that focus on some or all of the parameters. For example, a police officer may prefer more stable routes (low variability) in general, but at specific times it may need to find a faster route even though it is risky. In short, providing more options to users means that they can choose the most suitable one in each situation.

Figure 2 shows some sample Pareto fronts of RBSP and compares it with that of BSP. The solutions and Pareto front on the left consider that $\sigma^2 = 0$ for all the edges and the two parameters.

The solutions and Pareto front on the right assume that $\sigma^2 > 0$. Could make us find solutions with a low average cost, but with high variability (not very robust). However, if we expand the Pareto front to four dimensions, this will expand our set of non-dominated solutions with those that have low variability in our objectives.

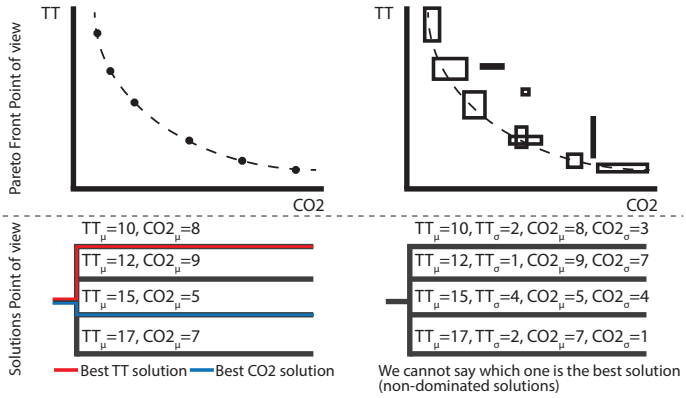


Figure 2: BSP problem (left) and RBSP problem (right) models. They are shown from the point of view of the Pareto fronts, as well as the selection of a edge when building the solution.

4. Experimentation Baseline

In this section, we describe the inputs, obtained from public open data websites, and algorithms used in our experimentation. With this section, we want to remark the applied intention of our work and the usability of our proposal in real-world applications.

4.1. Real Map

An essential part of the assistants for drivers is the road map of the city. We worked on the geographical region of the province of Malaga, Spain. This territory represents a medium size road map, as well as an European region. Its road map was obtained from the Open Street Map website[1], and processed by our own parser, that adapts the representation of a region to a graph, as we need in our formulation. Particularly, the road map of Malaga has 45,410 nodes and 118,388 edges. Figure 3 shows this graph.

We selected random points (according to a uniform distribution) to serve as the origin and destination nodes to the instances of the experimentation. We generated 29 instances in this way. All the data of the map and instances can be found in `http://neo.lcc.uma.es/staff/cintrano/research/RBSPproblem/RBSPproblem.html`.

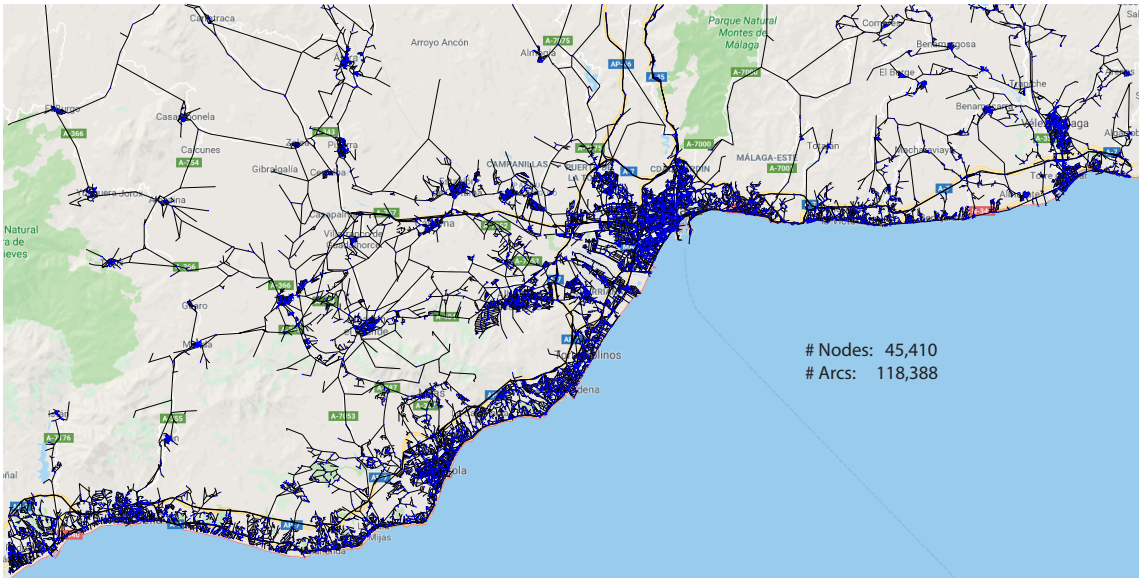


Figure 3: Graph over the map of the Malaga province, Spain

[1]Open Street Map website of Malaga city: `https://www.openstreetmap.org/#map=13/36.7248/-4.4253`

4.2. Realistic Inaccurate Parameters

As mentioned in Sec. 3, the two uncertain parameters considered in this work are travel time and gas emission. The mean of each parameter is computed for each edge of the graph as follows:

- Travel time (TT). For each road segment the mean travel time was calculated from the speed and distance obtained from Open Street Maps during the parse process of the source map.
- Gas emission (CO_2 and NO_x). We measure the gas emitted using the HBEFA model [21] and the estimated speed obtained from Open Street Maps. This model estimates the pollutant emission rate using the speed of the vehicle as follows:

$$h(v) = \max\left(0, c_0 + c_1 v\frac{dv}{dt} + c_2 v\left(\frac{dv}{dt}\right)^2 + c_3 v + c_4 v^2 + c_5 v^3\right), \tag{5}$$

  where the coefficients c_i with $0 \leq i \leq 5$ depend on the car and the type of gas emission. These coefficients are calculated as the result of averaging measurements of hundreds of types of vehicles, models, engines, etc.

The mean of the inaccurate parameters takes into account the real speed limits on roads and a commercial sedan as the vehicle for our study. We selected these specific values as an example of a possible driver of the city. Factors like traffic jams, traffic lights, works, etc. can also affect each parameter. In this work, the variances in both parameters were randomly generated. However, our model is prepared to consider the inaccuracies due to all these factors too. According to these real data, the fitness function used by our solvers is $\mathbf{z}^R(p)$ (see Equation 4). This function has four objectives: the means and the variances of the travel time and gas emission.

4.3. Optimization Algorithms

We choose five algorithms to solve the problem with different search strategies. The selected algorithms are Dijkstra algorithm [9], A* [19], Pulse [10], NSGA-II [8], and MOEA/D [39]. The first two algorithms are classic exact algorithms used to solve the shortest path problem in a graph. Pulse is a new exact algorithm designed to solve the multi-objective shortest path problem. NSGA-II and MOEA/D are popular multi-objective metaheuristic algorithms. All of them return paths between two single nodes s and e. These nodes and the graph are the inputs for the solvers. In the following paragraphs we provide a brief description of each of them.

Dijkstra shortest path algorithm is a well-known exact algorithm to find the shortest path between two nodes in a weighted graph. Its pseudocode is provided in Algorithm 1. Despite being a classic algorithm, it is still used today as a baseline in the design of new optimization strategies for the shortest path problem [32, 35].

A* (see Algorithm 2) is an algorithm based on Dijkstra's algorithm. It use of a heuristic function to estimate the cost of reaching the destination node from the current one. The heuristic function used in our study is the Euclidean distance between the geographical position of the nodes.

These two algorithms (Dijkstra and A*) are single-objective. We chose them because they are typically used in applications and state-of-the-art comparisons [1, 32, 37]. But, we are solving a multi-objective problem. So, we transform the problem from multi-objective to single-objective using a weighed sum of the objectives. This method allows us to find supported solutions from the Pareto front. The supported solutions are those belonging to the convex hull of the front. The transformed problem is as follows:

$$\min_{p \in \mathcal{P}_{s,e}} w(p) = \sum_{(i,j)\in p} (w_0\mu_{ij,1} + w_1\mu_{ij,2} + w_2\sigma^2_{ij,1} + w_3\sigma^2_{ij,2}) \tag{6}$$

with $w_i,\ 0 \leq i \leq 3$ the weights associated to each objective.

For each weight, we considered the following weight values: {0.0001, 0.25, 0.50, 0.75, 1}. Weight 0.0001 is used instead of 0 to ensure that the solution found is efficient (and not only weakly efficient). We execute the single-objective algorithm (Dijkstra or A*) for each combination of weight values. This means a total of 625 runs of the single-objective algorithms per instance (all combinations of weight values for the four weights: 5^4). With these weights evenly distributed in the interval (0, 1] we try to explore uniformly the objective space with these mono-objective algorithms. After the runs, we collect the obtained solutions to find a subset of the Pareto optimal set. In this paper, we denote with WDijkstra and WA* the Dijkstra's algorithm and A* solving the RBSP problem using this approach, respectively.

Algorithm 1 Dijkstra shortest path algorithm

Input a graph $G(N, A)$, start node s, and end node e

function DIJKSTRA'S ALGORITHM(s, e)
  for all $n \in N$ **do**
    $dist[n] \leftarrow \infty$
    $prev[n] \leftarrow null$
  $dist[s] \leftarrow 0$
  $Q \leftarrow N$
  while $Q \neq \emptyset$ **do**
    $u \leftarrow v \in Q$ with minimum $dist[v]$
    $Q \leftarrow Q \setminus u$
    if $u \neq e$ **then**
      for all neighbor v of u **do**
        $alt \leftarrow dist[u] + c_{uv}$
        if $alt < dist[v]$ **then**
          $dist[v] \leftarrow alt$
          $prev[v] \leftarrow u$
  $S \leftarrow []$
  $u \leftarrow e$
  while $prev[u]$ is not null **do**
    $S \leftarrow u \cdot S$
    $u \leftarrow prev[u]$
  $S \leftarrow u \cdot S$
  return S

Algorithm 2 A* algorithm

Input a graph $G(N, A)$, start node s, and end node e

function A* ALGORITHM(s, e)
  $closed \leftarrow \emptyset$
  $open \leftarrow \{s\}$
  $cameFrom \leftarrow$ empty map
  $gScore \leftarrow$ map with default value of ∞
  $gScore[s] \leftarrow 0$
  $fScore \leftarrow$ map with default value of ∞
  $fScore[s] \leftarrow h(s, e)$
  while $open \neq \emptyset$ **do**
    $u \leftarrow n \in open$ with the lowest $fScore[n]$
    if $u = e$ **then**
      $path \leftarrow [u]$
      while $u \in cameFrom.Keys$ **do**
        $u \leftarrow cameFrom[u]$
        $path \leftarrow u \cdot path$
      return totalPath
    $open \leftarrow open \setminus \{u\}$
    $closed \leftarrow closed \cup \{u\}$
    for all neighbors v of u **do**
      if $v \notin closed$ **then**
        if $v \notin open$ **then**
          $open \leftarrow open \cup \{v\}$
        if $gScore[u] + c_{uv} < gScore[v]$ **then**
          $cameFrom[v] \leftarrow u$
          $gScore[v] \leftarrow gScore[u] + c_{uv}$
          $fScore[v] \leftarrow gScore[v] + h(v, e)$
  return failure

The Pulse algorithm, especially its version for more than two objectives, was proposed by Duque et al. [10]. It is an exact method to solve the multi-objective shortest path problem. Pulse uses a strong pruning strategy to discard branches that are dominated by the solutions found so far. The set of solutions is updated when the algorithm obtains new non-dominated solutions. This allows us to get different sub-optimal Pareto sets (approximations of the optimal Pareto set) during the execution of the algorithm. The main pseudocode is presented in Algorithm 3.

Algorithm 3 Multi-objective Pulse algorithm

Input a graph $G(N, A)$, start node s, and end node e

function PULSE ALGORITHM(s, e)
  $P \leftarrow \{\}$
  $\vec{c}(P) \leftarrow 0$
  initialization(G)
  Pulse($s, \vec{c}(P), P$)
  return X_E
function PULSE(v_i, $\vec{c}(P)$, P)
  if isAcyclic(v_i, P) **then**
    if $\neg checkNadirPoint(v_i, \vec{c}(P))$ **then**
      if $\neg checkEfficientSet(v_i, \vec{c}(P))$ **then**
        if $\neg checkLabels(v_i, \vec{c}(P))$ **then**
          store($\vec{c}(P)$)
          $P' \leftarrow P \cup \{v_i\}$
          for $v_j \in$ outgoingNeighbors(v_i) **do**
            for $k = 1$ to $size(\vec{c})$ **do**
              $c_k(P') \leftarrow c_k(P) + c_{ij,k}$
            Pulse($v_j, \vec{c}(P'), P'$)

NSGA-II is a genetic algorithm proposed by Deb et al. [8]. The pseudocode is described in Algorithm 4. It is commonly used by the multi-objective researchers [3].

Algorithm 4 NSGA-II algorithm

Input a graph $G(N, A)$, start node s, and end node e

function NSGA-II ALGORITHM(s, e)
  $P \leftarrow initializePopulation()$
  $PF \leftarrow \emptyset$
  while not stopping condition **do**
    $Q = \emptyset$
    for $i = 1$ to popSize **do**
      $parents \leftarrow selection(P)$
      $child \leftarrow crossover(parents)$
      $child \leftarrow mutation(child)$
      $Q = Q \cup \{child\}$
    $P' = Q \cup P$
    reportNonDominaedSolutions(PF, P')
    $P = selectUsingRankingAndCrowding(P')$
  return P

MOEA/D is an evolutionary algorithm proposed by Zhang and Li [39]. In this algorithm, we have used the proposed version called MOEA/D-STM [28]. The pseudocode of this version is presented in Algorithm 5.

NSGA-II and MOEA/D have some common configuration parameters. In order to do a fair comparison among them we have used the same parameters in both algorithms. These are:

- Selection operator: random selection.
- Crossover operator: one-point crossover with probability 0.9.
- Mutation operator: custom mutation with probability 0.1. This mutation selects a random middle point k in a (s-e)-path and a random vertex i and compute the two paths executing the A* algorithm: (k-i)-path and (i-e)-path. After computing the shortest path between the pairs (s-k), (k-i), and (i-e), it joins them to form the new solution path (s-k-i-e)-path.
- Replacement operator: elitist tournament.
- Neighbourhood of 5 individuals (only MOEA/D).

The stopping condition in NSGA-II and MOEA/D is to reach 10,000 fitness evaluations. The population size is 10 individuals.

Algorithm 5 MOEA/D algorithm

Input a graph $G(N, A)$, start node s, and end node e

function MOEA/D-STM ALGORITHM(s, e)
  $population \leftarrow initializePopulation()$
  $weightVectors \leftarrow initializeWeightVectors()$
  $\vec{z}^* \leftarrow setIdealVector()$
  $\vec{z}^{nad} \leftarrow setNadirVector()$
  for $index \in \{1,\ maxGenerations\}$ **do**
    $B(i) \leftarrow \{i_1, \ldots, i_T\}$ *were* $\vec{w}^{i_1}, \ldots, \vec{w}^{i_T}$ *are the* T *closest weight vectors to* $\vec{w}^i$
    $\pi^i \leftarrow 1$
  for $i \in \{1,\ maxGenerations\}$ **do**
    $Q \leftarrow \emptyset$
    for *each* $i \in I$ **do**
      if $uniform(0,1) < \delta$ **then** $E \leftarrow B(i)$
      else $E \leftarrow P$
      $newPopulation \leftarrow generateChildPopulation(poulation)$
      $parents \leftarrow selection(newPopulation)$
      $child \leftarrow crossing(parents)$
      $child \leftarrow mutate(child)$
      $Q \leftarrow Q \cup child$
      $update(\vec{z}^*)$
      $update(\vec{z}^{nad})$
    $R \leftarrow population \cup Q$
    $population \leftarrow STM(R, W, \vec{z}^*, \vec{z}^{nad})$
  return population

4.4. Hardware Platform

The computation platform used in this work is composed of a cluster of 96 cores, equipped with two Intel Xeon CPU (E5-2670 v3) at 2.30 GHz and 64 GB memory. The cluster is managed by HTCondor 8.2.7, which allows us to perform parallel independent executions to reduce the overall execution time. The implementation was done in Java 7 programming language on a machine with Ubuntu 16.04 LTE.

5. Experimental Results

In this section we will present the main results of our study. First, we will analyze the Pareto fronts obtained, followed by the robustness of the different solutions found for each instance. Finally, the algorithms WDijkstra and WA* will be specifically analyzed apart, focusing on the relationship between the weights and the fitness values for the different solutions found.

5.1. Pareto Optimal Set Analysis

In this section we analyze the quality of the approximated Pareto optimal set found by the algorithms in the different instances for two different types of polluting gases: CO_2 (29 instances) and NO_x (28 instances). NSGA-II and MOEA/D got solutions which do not reach the same quality as the exact algorithms in all the experiments. In fact, the quality of their solutions with a similar execution time was much lower than the rest of the proposals. They did not get any solution from the Pareto front in any of the instances. That is why they are not explained in detail in this section. This behavior makes these metaheuristic algorithms a bad option for solving this problem.

5.1.1. Travel times and CO_2

Table 1 reports the execution time per solution and size of the sets of non-dominated solutions obtained by the deterministic algorithms when they analyze the travel times and CO_2. Pulse returns the entire front, so the reported time t is the total running time of the algorithm T divided by the number n of solutions in the Pareto front: $t = T/n$. An interesting result is that WDijkstra and WA* obtained the same results for each instance, but WA* is much faster, as expected. The Pulse algorithm returned the whole Pareto front at a cost of a higher run time.

Figure 4 shows the time per solution, in milliseconds, for each algorithm and instance. The running times follow ascending lines. However, a higher computation time does not imply a larger Pareto set. WDijkstra had stable behavior in their running time while WA* was a little less stable.

The sizes of the non-dominated sets are shown in Figure 5. The percentage of solutions found in the optimal front by WDijkstra and WA* is 17.77 % on average. This percentage suggests that the

Table 1: Runtime to compute one solution for WDijkstra, WA* and Pulse and number of different solutions found in each of the TT-CO_2 instances. We mark the shortest time and largest number of solutions found for each instance.

Instance	Start	End	Runtime per solution (ms)			# of solutions		
			WDijkstra	WA*	Pulse	WDijkstra	WA*	Pulse
1	32310	27542	1801	55	53398	4	4	48
2	12175	40684	1743	732	72523	4	4	11
3	40684	12175	1375	756	77136	3	3	9
4	27542	32310	1936	45	77957	5	5	42
5	13360	10386	1677	543	113866	6	6	23
6	30159	8481	2032	324	132024	13	13	157
7	23385	11669	1731	225	159944	10	10	49
8	10789	11798	1361	343	178261	4	4	70
9	11669	23385	1771	178	180364	6	6	20
10	28960	28582	1534	284	182434	7	7	86
11	39559	20777	2000	63	188075	7	7	63
12	10386	13360	1208	138	211592	6	6	16
13	11798	10789	2205	643	257288	5	5	57
14	42100	21155	1261	291	261992	6	6	161
15	15181	30200	2060	531	279094	9	9	77
16	14141	33551	1377	371	302760	5	5	223
17	33551	14141	1559	294	308817	5	5	136
18	8481	30159	1630	192	343680	6	6	82
19	19215	39301	1405	120	406114	5	5	10
20	39301	19215	1062	87	437103	4	4	27
21	20777	39559	1771	260	470971	8	8	17
22	21155	42100	2056	862	476278	4	4	80
23	45081	34504	1139	215	778485	12	12	263
24	30200	15181	1184	292	854915	15	15	168
25	34504	45081	1474	471	872470	7	7	88
26	28582	28960	1554	183	1200626	8	8	28
27	25319	35869	1525	591	2286564	10	10	27
28	20842	9585	2130	747	2312745	5	5	206
29	35869	25319	1263	210	4591774	11	11	32
Total			46824	10046	18069250	200	200	2276

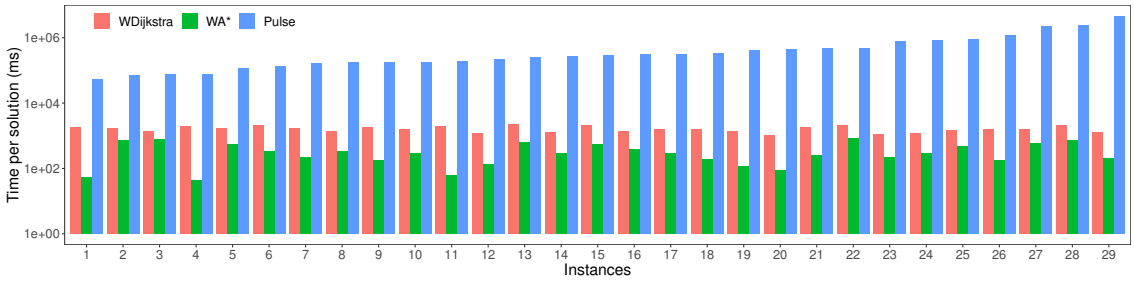


Figure 4: Execution time per solution (ms) for the execution of each deterministic algorithm in every instance of the problem with CO_2 as pollutant.

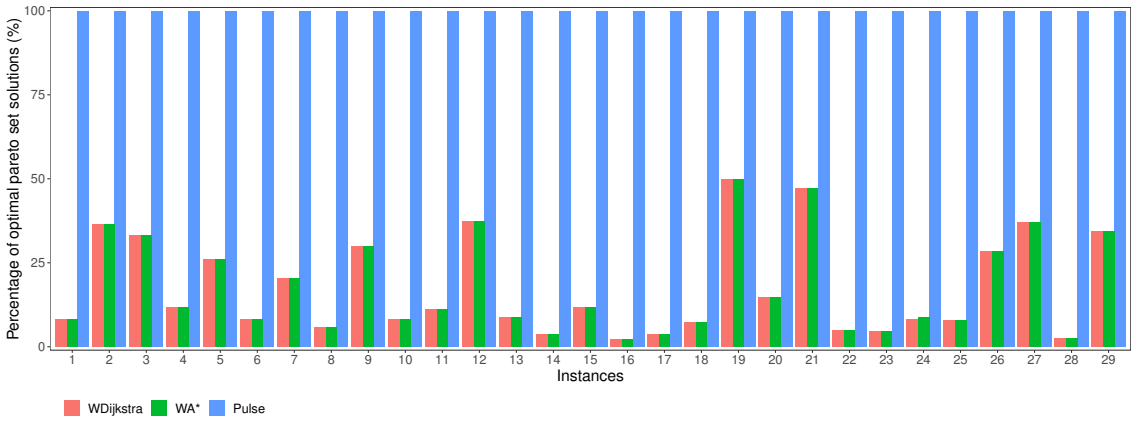


Figure 5: Percentage of the optimal Pareto set found for each deterministic algorithm in every instance of the problem with CO_2 as pollutant.

number of supported solutions in the front is low. Despite the 625 weight combinations computed in WDijkstra and WA*, the obtained non-dominated sets have between three and twelve solutions only. This is because each point in the target space has a polytope associated with it in the weight space and several weight combinations are in the same polytope. In Section 5.3 we will study the combinations of weights in more detail.

Now, we will analyze the quality of the approximated Pareto fronts by some quality indicators [34]. We select as metrics the hypervolume, ϵ-indicator, and inverse generational distance because are commonly used in the multi-objective literature. Their values are in Table 2. Figure 6 shows the hypervolume percentage in each instance. We can see that, in some cases, the hypervolume of the deterministic algorithms are very close to each other (especially in instance 29). NSGA-II has better hypervolume than MOEA/D, even getting close to the deterministic algorithms. In general, both metaheuristics offer worse hypervolumes, except in instance 28, in which

Table 2: Quality indicators of the approximated Pareto fronts in each instance for the TT-CO_2 case. The highest values for the hypervolume and the lowest values for ϵ-indicator and inverse generational distance are highlighted.

Ins.	Hypervolume					ϵ-indicator					Inverse generational distance				
	WD.	WA*	Pulse	NSGA-II	MOEA/D	WD.	WA*	Pulse	NSGA-II	MOEA/D	WD.	WA*	Pulse	NSGA-II	MOEA/D
1	6.84E+18	6.84E+18	6.94E+18	6.51E+18	5.52E+18	282	282	0	218	26135	14695	14695	0	1867	14663
2	1.09E+18	1.09E+18	1.09E+18	1.07E+18	1.07E+18	41	41	0	46	64	3388	3388	0	7492	2030
3	1.11E+18	1.11E+18	1.11E+18	1.09E+18	1.09E+18	59	59	0	64	64	1496	1496	0	11158	3920
4	6.82E+18	6.82E+18	6.91E+18	6.49E+18	5.48E+18	271	271	0	216	26244	13838	13838	0	9453	17030
5	1.12E+18	1.12E+18	1.12E+18	1.12E+18	1.12E+18	117	117	0	20	20	4726	4726	0	1890	1534
6	2.33E+18	2.33E+18	2.34E+18	2.28E+18	2.25E+18	137	137	0	341	1292	6927	6927	0	1904	2367
7	2.26E+18	2.26E+18	2.26E+18	2.24E+18	2.22E+18	42	42	0	346	665	5654	5654	0	3543	8213
8	1.73E+18	1.73E+18	1.74E+18	1.71E+18	1.70E+18	112	112	0	724	724	17597	17597	0	2757	1506
9	2.33E+18	2.33E+18	2.33E+18	2.31E+18	2.30E+18	28	28	0	422	422	2692	2692	0	7775	3284
10	2.12E+18	2.12E+18	2.15E+18	2.13E+18	1.78E+18	489	489	0	145	509	8673	8673	0	1500	4729
11	3.46E+18	3.46E+18	3.50E+18	3.44E+18	3.21E+18	334	334	0	951	951	11747	11747	0	1393	13320
12	1.08E+18	1.08E+18	1.08E+18	1.08E+18	1.08E+18	68	68	0	8	15	2280	2280	0	3087	2521
13	1.73E+18	1.73E+18	1.74E+18	1.73E+18	1.71E+18	103	103	0	37	145	12218	12218	0	2561	2733
14	7.72E+16	7.72E+16	7.81E+16	7.70E+16	7.66E+16	69	69	0	420	420	7980	7980	0	6081	17281
15	2.71E+17	2.71E+17	2.73E+17	2.70E+17	2.58E+17	140	140	0	961	961	8282	8282	0	30696	44340
16	2.63E+18	2.63E+18	2.67E+18	2.61E+18	2.56E+18	157	157	0	104	164	26941	26941	0	47710	53018
17	2.74E+18	2.74E+18	2.80E+18	2.73E+18	2.68E+18	183	183	0	118	168	27166	27166	0	15004	91262
18	2.35E+18	2.35E+18	2.36E+18	2.30E+18	2.27E+18	135	135	0	313	313	7628	7628	0	1680	2951
19	4.04E+18	4.04E+18	4.05E+18	4.03E+18	4.00E+18	94	94	0	398	398	5441	5441	0	1939	2555
20	4.08E+18	4.08E+18	4.09E+18	4.07E+18	3.97E+18	92	92	0	61	102	12275	12275	0	11908	3205
21	3.56E+18	3.56E+18	3.57E+18	3.55E+18	3.21E+18	144	144	0	144	4758	1457	1457	0	7211	10852
22	8.58E+16	8.58E+16	8.68E+16	8.58E+16	8.56E+16	70	70	0	62	66	7611	7611	0	4015	12351
23	4.92E+17	4.92E+17	5.05E+17	4.85E+17	4.72E+17	251	251	0	999	999	8523	8523	0	6371	41705
24	2.77E+17	2.77E+17	2.79E+17	2.78E+17	2.66E+17	144	144	0	33	167	6226	6226	0	13961	43379
25	7.33E+17	7.33E+17	7.41E+17	7.18E+17	6.91E+17	196	196	0	842	842	17948	17948	0	12639	33830
26	2.11E+18	2.11E+18	2.11E+18	2.10E+18	1.68E+18	444	444	0	444	1229	644	644	0	638	2086
27	7.84E+17	7.84E+17	7.84E+17	7.80E+17	7.42E+17	34	34	0	160	530	2079	2079	0	1448	7484
28	1.48E+17	1.48E+17	2.11E+17	1.95E+17	1.82E+17	2249	2249	0	1159	1403	22209	22209	0	1569	6347
29	7.02E+17	7.02E+17	7.02E+17	6.96E+17	6.46E+17	11	11	0	1109	1109	3802	3802	0	2238	4437

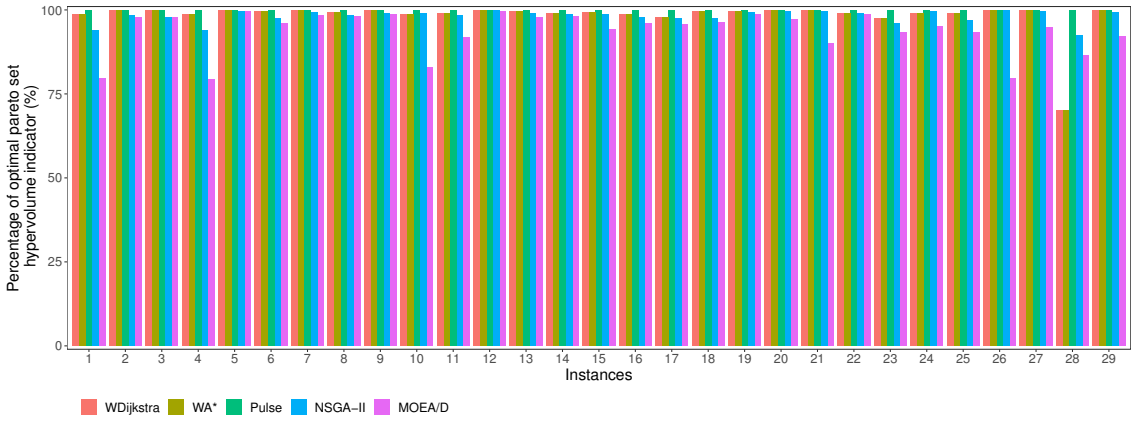


Figure 6: Percentage of the hypervolume indicator of the approximated optimal Pareto set for the execution of each algorithm in every instance of the problem with CO_2 as pollutant.

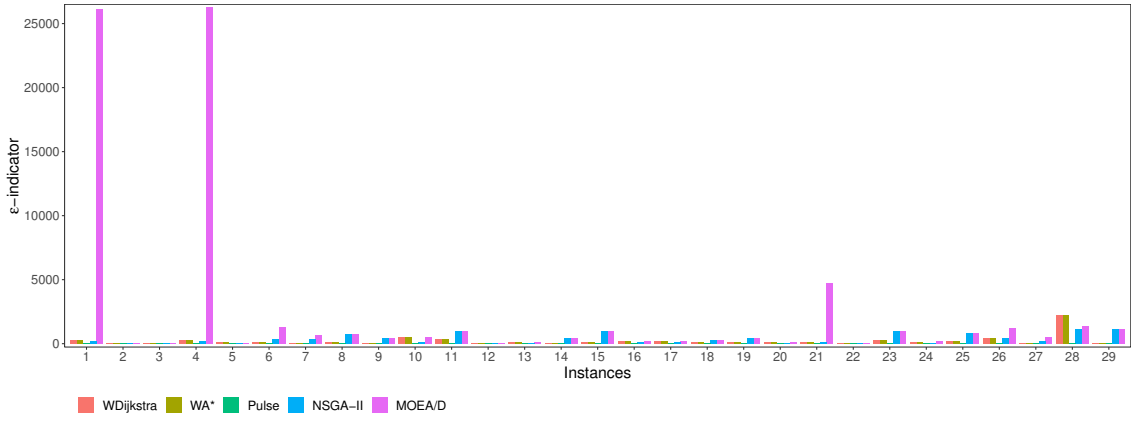


Figure 7: ϵ-indicator for the execution of each algorithm in every instance of the problem with CO_2 as pollutant.

the few solutions obtained by WDijkstra and WA* are not enough to obtain a high hypervolume. This is a good result for these solvers, because only with the supported solutions we can cover very well the objective space in the general case. The results for the number of non-dominated solutions and the hypervolume suggest that many of the solutions are non-supported and their contribution to the hypervolume is low.

If we see the ϵ-indicator (see Figure 7) of each algorithm (where lower values are better), we will find that in some instances the distance of the approximated Pareto front found by WDijkstra and WA* is very close (11 units) to the optimal front, and in general it is not so far from the Pareto front found by the Pulse algorithm. MOEA/D obtained the worst measures for this indicator especially

Table 3: Runtime to compute one solution for each deterministic algorithm in each instance and number of different solutions found in the TT-NO_x case. We mark the shortest time and largest Pareto front size for every instance.

Instance	Start	End	Runtime per solution (ms)			# of solutions		
			WDijkstra	WA*	Pulse	WDijkstra	WA*	Pulse
1	32310	27542	2702	44	48945	9	9	42
2	12175	27543	2007	1310	16081	8	8	12
3	40684	27544	1323	389	659375	8	8	34
4	27542	27545	2351	32	57746	9	9	35
5	13360	27546	1703	1301	66766	6	7	16
6	30159	27547	3467	415	188996	36	36	176
7	23385	27548	2404	522	85042	22	22	73
8	10789	27549	2963	772	68870	11	12	77
9	11669	27550	2011	469	112745	11	11	27
10	28960	27551	2093	1353	101731	4	4	62
11	39559	27552	2520	101	332607	11	11	52
12	10386	27553	1508	184	142696	7	7	17
13	11798	27554	4309	710	75567	12	13	107
14	42100	27555	1227	338	203377	27	28	152
15	15181	27556	1550	597	135527	16	17	40
16	14141	27557	3516	528	154292	15	16	208
17	33551	27558	1862	824	242193	14	14	82
18	8481	27559	1498	131	250644	13	13	55
19	19215	27560	3340	247	185996	4	4	7
20	39301	27561	3350	133	283093	4	5	11
21	20777	27562	2608	506	196655	8	8	12
22	21155	27563	2488	1296	190553	14	14	52
23	45081	27564	909	265	376058	25	26	273
24	30200	27565	819	265	397895	24	24	145
25	34504	27566	1825	813	447447	11	11	54
26	28582	27567	2089	335	257131	5	5	96
27	25319	27568	1812	739	2901262	7	7	12
29	35869	27570	2408	422	6811269	9	10	27
Total			65128	15639	14990559	372	381	1956

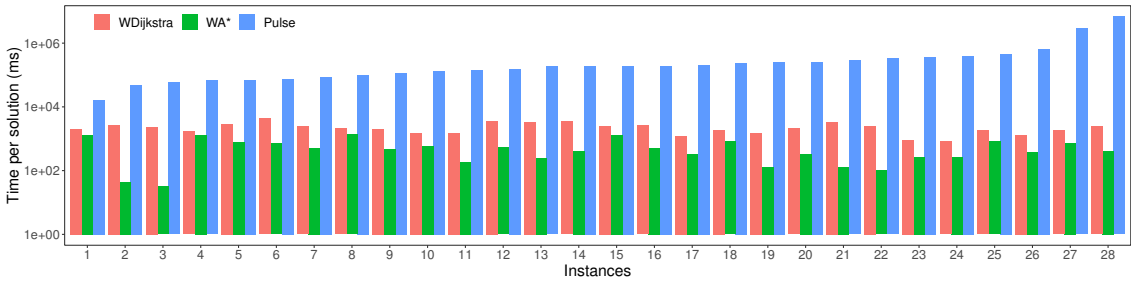


Figure 8: Execution time per solution (ms) for the execution of each deterministic algorithm in every instance of the problem with NO_x as pollutant.

in three instances. This is an advantage of using single-objective algorithms and weighed sum to get an approximation of the Pareto front in the RBSP problem, since a very fast approximated front is obtained with an acceptable quality, even discarding the non-supported solutions.

5.1.2. Travel times and NO_x

In this section, we repeat the previous analysis replacing the CO_2 by NO_x. Table 3 reports the running time per solution and the number of them found by WDijkstra, WA*, and Pulse. While the execution times are similar to those of the version with CO_2 (compare Figures 4 and 8) the number of found solutions differs. The number of solutions obtained by the strategy of weighted sums is greater than in the case of CO_2. In general, 86% more solutions have been found in fronts that have 14% fewer solutions, so the approximations are generally better than with CO_2 (see Figure 9). It should also be noted that in nine cases WA* found one solution more than WDijkstra.

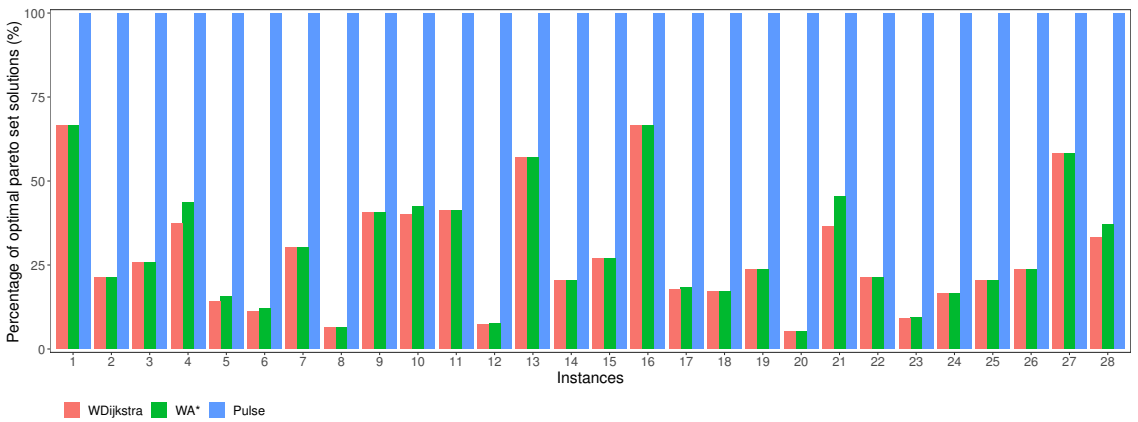


Figure 9: Percentage of the optimal Pareto set found for each deterministic algorithm in every instance of the problem with NO_x as pollutant.

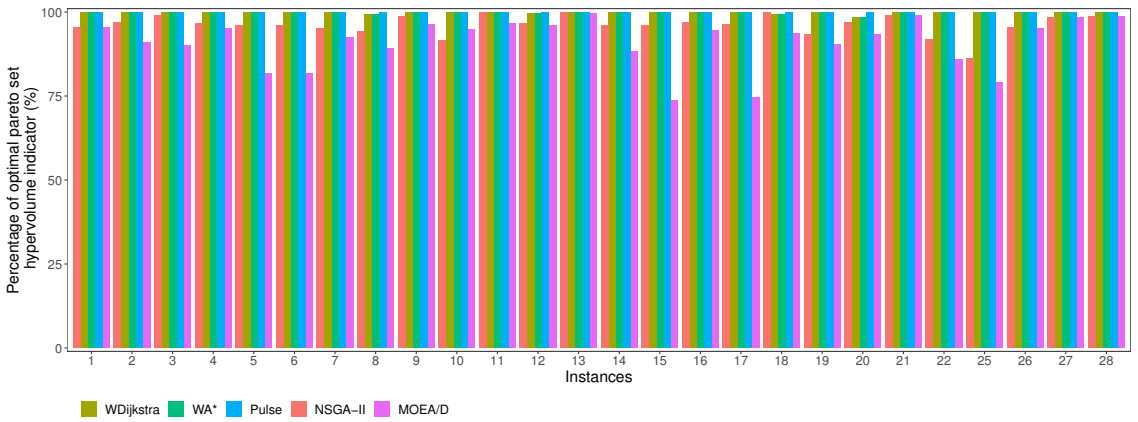


Figure 10: Percentage of the hypervolume indicator of the approximated optimal Pareto set for the execution of each algorithm in every instance of the problem with NO_x as pollutant.

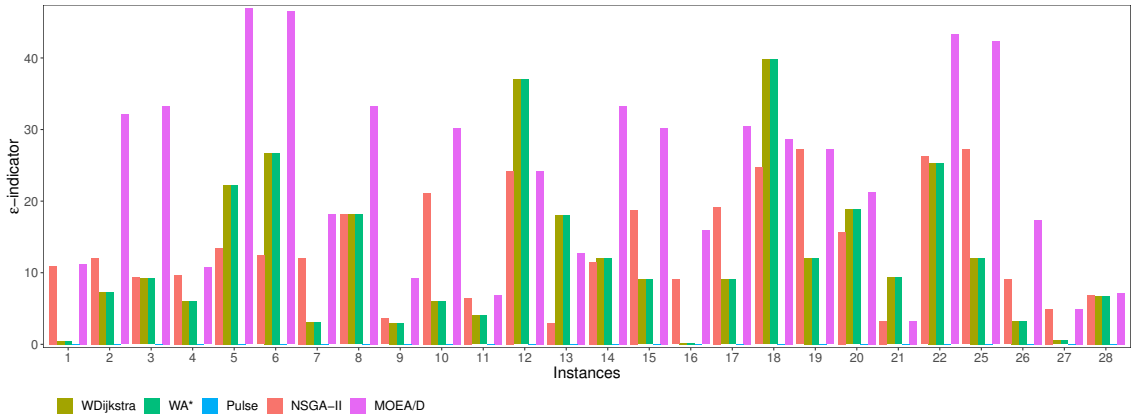


Figure 11: ϵ-indicator for the execution of each algorithm in every instance of the problem with NO_x as pollutant.

We can state that the problem has more supported solutions as NO_x is considered, compared to the case of CO_2. If the number of solutions and the runtime are important in an application, this result suggests that NO_x should be considered in the optimization.

Regarding the quality indicators of the different instances, shown in Table 4, the results in the three metrics outperform those obtained in the case of CO_2. In the case of the hypervolume (see Figure 10) we observe once again that the three deterministic algorithms obtain a similar value and the two metaheuristic algorithms have the lowest performance, in general. Figure 11 shows the results of the ϵ-indicator, which also significantly improve the ones obtained for CO_2.

Table 4: Quality indicators of the Pareto fronts in each instance of the TT-NO_x case. The highest values for the hypervolume and the lowest values for ϵ-indicator and inverse generational distance are marked.

Ins.	Hypervolume					ϵ-indicator					Inverse generational distance				
	WD.	WA*	Pulse	NSGA-II	MOEA/D	WD.	WA*	Pulse	NSGA-II	MOEA/D	WD.	WA*	Pulse	NSGA-II	MOEA/D
1	1.49E+13	1.49E+13	1.49E+13	1.45E+13	1.36E+13	7	7	0	12	32	14	14	0	17	33
2	3.06E+12	3.06E+12	3.06E+12	2.92E+12	2.92E+12	0	0	0	11	11	2	2	0	14	19
3	3.07E+12	3.07E+12	3.07E+12	2.93E+12	2.92E+12	3	3	0	9	17	5	5	0	15	23
4	1.49E+13	1.49E+13	1.49E+13	1.48E+13	1.34E+13	9	9	0	9	33	11	11	0	106	131
5	3.07E+12	3.07E+12	3.08E+12	2.97E+12	2.92E+12	6	6	0	10	11	13	13	0	15	27
6	5.43E+12	5.43E+12	5.44E+12	5.23E+12	4.80E+12	12	12	0	11	33	9	9	0	15	34
7	5.69E+12	5.69E+12	5.69E+12	5.41E+12	5.27E+12	3	3	0	12	18	3	3	0	212	176
8	4.22E+12	4.22E+12	4.23E+12	4.06E+12	3.45E+12	22	22	0	13	47	28	27	0	16	56
9	5.90E+12	5.90E+12	5.90E+12	5.82E+12	5.68E+12	3	3	0	4	9	3	3	0	9	19
10	4.97E+12	4.97E+12	5.01E+12	4.72E+12	4.46E+12	18	18	0	18	33	139	139	0	19	41
11	8.05E+12	8.05E+12	8.07E+12	7.41E+12	6.94E+12	25	25	0	26	43	51	51	0	14	32
12	3.06E+12	3.06E+12	3.06E+12	3.05E+12	2.95E+12	4	4	0	6	7	5	5	0	7	15
13	4.20E+12	4.20E+12	4.21E+12	4.04E+12	3.44E+12	27	27	0	12	46	29	29	0	12	49
14	5.71E+11	5.71E+11	5.72E+11	5.50E+11	4.26E+11	9	9	0	19	30	9	9	0	22	37
15	1.11E+12	1.11E+12	1.11E+12	1.01E+12	1.05E+12	6	6	0	21	30	5	5	0	27	30
16	6.16E+12	6.16E+12	6.18E+12	5.97E+12	5.94E+12	37	37	0	24	24	50	37	0	29	28
17	6.38E+12	6.38E+12	6.42E+12	6.41E+12	6.01E+12	40	40	0	25	29	52	52	0	31	97
18	5.43E+12	5.43E+12	5.43E+12	5.06E+12	4.91E+12	12	12	0	27	27	9	9	0	40	29
19	8.58E+12	8.58E+12	8.60E+12	8.59E+12	8.57E+12	18	18	0	3	13	15	15	0	15	11
20	8.69E+12	8.69E+12	8.69E+12	8.61E+12	8.61E+12	9	9	0	3	3	12	8	0	41	95
21	8.17E+12	8.17E+12	8.17E+12	7.92E+12	7.72E+12	0	0	0	9	16	5	5	0	38	30
22	5.64E+11	5.64E+11	5.65E+11	5.42E+11	4.15E+11	9	9	0	19	30	9	9	0	12	17
23	1.71E+12	1.71E+12	1.72E+12	1.40E+12	1.25E+12	11	11	0	33	51	11	11	0	51	51
24	1.13E+12	1.13E+12	1.13E+12	1.12E+12	1.06E+12	9	9	0	9	34	12	12	0	23	72
25	2.19E+12	2.19E+12	2.19E+12	1.89E+12	1.74E+12	12	12	0	27	42	14	14	0	67	55
26	4.87E+12	4.87E+12	4.94E+12	4.78E+12	4.62E+12	19	19	0	16	21	263	263	0	64	85
27	2.28E+12	2.28E+12	2.28E+12	2.24E+12	2.24E+12	1	1	0	5	5	2	2	0	26	78
29	2.15E+12	2.15E+12	2.15E+12	2.13E+12	2.13E+12	7	7	0	7	7	7	6	0	4	5

In these two quality indicators, there are no differences between WDijkstra and WA*. However, in the inverse generational distance, there are differences in some cases. Although WA* improved the number of solutions in 9 scenarios but the metric is only improved in 3 of them. This suggests that, despite good approximations are obtained, there may still be regions in which there are supported solutions that have not been found. Metaheuristics continue to get the worst results.

5.2. Robustness Analysis

In this section, we analyze the degree of robustness of the solutions. A difference in our proposal compared to others is that we offer a set of solutions (not only one) with varying levels of quality and robustness in each objective (travel time and gas emission). As an example, Figure 12 shows the Pareto front of instance 13 optimized for TT and CO_2. The colors represent the different algorithms used in this study, and the size of the boxes represent the robustness of a solution (its standard deviation). Each solution is represented by a box which is centered in the mean CO_2 emission and travel time, and its width and height is the standard deviation for CO_2 emission and travel time, respectively. Thus, a very robust solution has a small box, while a large box indicates that the solution quality has a large variability in their objectives (TT or CO_2).

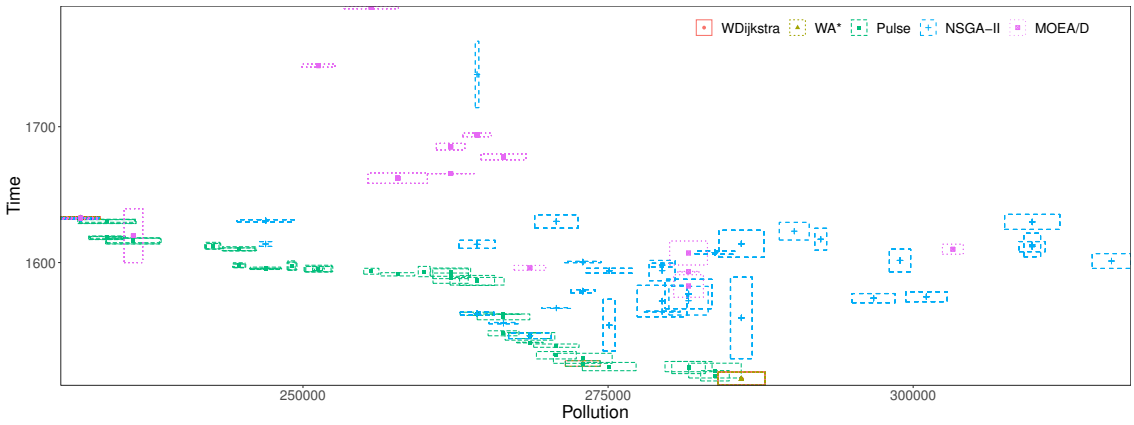


Figure 12: Approximated Pareto fronts of instance 13 for TT-CO_2 found by the each algorithm. The dimension of the squares is the standard deviation for the travel time and CO_2 emission, respectively.

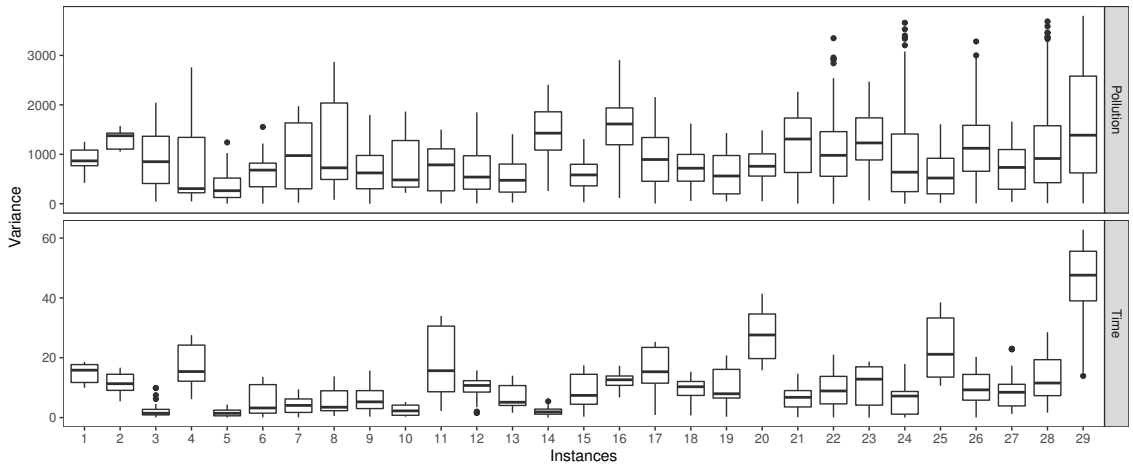
We can observe that a large part of the solutions (obtained by Pulse) is in the concave part of the front. These solutions cannot be found by WDijkstra and WA* algorithms. It can also be seen how the solutions that in a non-robust version would be dominated, have a smaller variance (smaller box) in the objectives. On the other hand, NSGA-II and MOEA/D do not reach the level of quality of the deterministic algorithms, obtaining solutions very far from the Pareto front.

In most of the cases, the variation in the objectives between the most and least robust solution is small. But this does not mean that robustness does not have to be taken into account, because the area of the box is not of the same size (a small box represent a most robust solution). Although in most cases the squares are the same size, in each front there is a subset whose size is smaller.

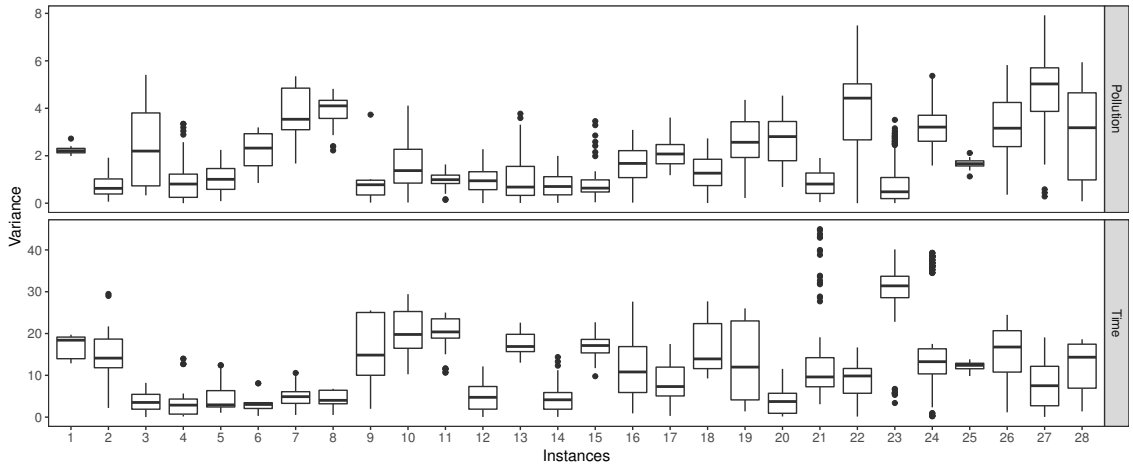
Now, we analyze in more depth the approximated Pareto fronts obtained by Pulse. In Figure 13, we show the variance of the pollution gases, CO_2 and NO_x, and TT among the different solutions in the Pareto front. We can see that the TT has a smaller variation than the CO_2, being especially small in some of the instances, that slightly increasing the user's travel times greatly reduces the levels of CO_2 emitted. This makes us very optimistic with our model, as it is interesting to explore solutions with low variation in objectives. However, as in instance 29, the variance changes a lot between the solutions, so there are cases in which TT are greatly affected by the chosen route. In the case of NO_x, it is observed that the variances are lower than in the case of CO_2. More robust solutions are obtained if NO_x is used as a pollutant gas. But, in general, more extreme solutions are observed compared to the CO_2 case.

5.3. Weights Analysis

The difference between the number of total weight combinations (625) and different efficient solutions (a few tens) found by the algorithms WDijkstra and WA* opens the possibility of studying which weights return the same solutions. Discovering these relationships between weights and fitness can help us to reduce the computation times of this strategy and allows us to discard weight combinations very quickly.



(a) Travel times and CO_2 case.



(b) Travel times and NO_x case.

Figure 13: Variances of each objective in relation to the mean for each instance in the Pulse algorithm.

Formally, a supported non-dominated solution in the Pareto front is associated to a polytope in the weights space, e.g., a set of weights values that return the same solution in the optimization process. We only need a single weight combination in the interior of each polytope to get the whole set of supported solutions associated with that polytope.

We will analyze one of the instances as an example. Figure 14 shows the different weights combinations in a single instance. In the plot, the colors indicate a specific solution obtained with the weight combination. Several boxes with the same color form a set of combinations of weights that generate the same solution. There are a pattern in the colored grids. When $w_3 > 0.0001$, the number of solutions found are considerably limited. This weight has a large influence on the exploration of the region where most of the supported solutions are located. Using two values for w_3 could make us reduce the computational time. Weights w_0, w_1 and w_2 do not have an direct influence in the supported solutions explored (as w_3 has): varying these three weights many polytopes defining different supported solutions are reached.

These results confirm our idea of reducing execution times by analyzing the weights. In our experimentation this would suppose a reduction of 98.47 %. We think that these patterns in the relationship between the weights and the solutions can be calculated from the characteristics of the problem (map and start and end points), but we defer this issue to future work.

6. Discussion

We review in this section some works related to the main topic of this research. Starting with the concept of robustness, Jin and Branke [24] defines it as the desirable characteristic of taking into account the uncertainties in different sources of information (parameters, data, etc.). One way of looking at robustness is considering the inaccuracies in the parameters of the problem that

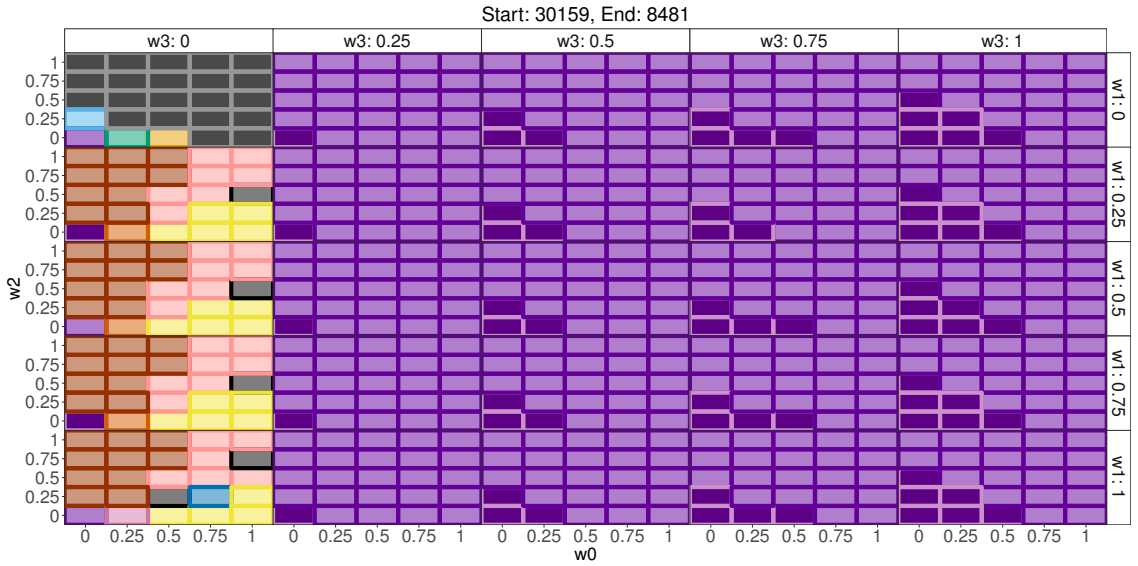


Figure 14: Weight combinations to obtain each supported solution (different colour) in the instance number 6 optimized for TT and CO_2.

come from multiple sources, such as approximate data or incorrect sensor measures. Usually, the inaccuracies can be defined in different ways:

- The parameter p can have values from a finite and defined set of values $p \in S = \{s_1, s_2, \ldots, s_n\}$. Each element in the set is called a *scenario* [27].
- The parameters can take values from an infinite set, e.g., real numbers, $p \in [a, b]$. Because the techniques for solving them are often similar, this group also includes probabilistic functions and confidence intervals [11, 13, 16], among others.

The type of values in the parameters has a direct implication in the chosen model of robustness and in the techniques to solve the robust problem. Our proposal is included in the second definition of inaccuracies in the parameters.

There are several surveys that analyze distinct aspects of robustness. An overview of the whole paradigm of robustness is described in the survey of Gabrel et al. [13]. This work defines and lists most of the scientific works done in robust optimization (in its general meaning) in the last years. In [23], a classification of different types of robust problems is presented. The robust solutions are classified according to the number of scenarios in which the solution is optimal. In this way, one solution is more robust than another when it is the optimal solution in a larger number of scenarios. But, it is not always possible to have well-defined scenarios. Moreover, not all scenarios are the same or have the same relevance. In this taxonomy, the minmax strategy is adapted for each kind of robustness. The minmax strategy [26] tries to minimize the following expression:

$$\min_{x \in X} \max_{s \in S} f_s(x) \tag{7}$$

where S is the set of scenarios, X the solution space, and $f_s(x)$ the objective function in the scenario s. The solution x is an upper bound for all the scenarios. Because of this, it is very common to use the minmax strategy when solving the shortest path problem with scenarios [32]. However, fairly considers usual and unusual scenarios. In many cases, users may prefer to lose a bit of robustness in exchange of a significant improvement in the quality of the solution.

If the robustness cannot be modeled in a discrete way, a range of possibilities opens up: fuzzy logic [25], random delays variables [6], confidence intervals [17, 18, 20, 38], probabilistic distributions [4], etc. The minmax strategy can also be applied when the data can change in a defined interval $[a, b]$. This limitation in the decision variables can be used by a minmax strategy to minimize the worst case (upper bound of each interval) scenario, as done in [22, 33].

Besides saying whether a solution is robust or not, a certain degree or level of robustness could be given to a single solution. According to the classification that is presented in [27], we can assign a level of robustness to a solution, i.e., each robust definition of a problem have an associated level of robustness for its solutions. A common way to measure the robustness is by a confidence ratio [30], a.k.a. radius of robustness [15]. It defines a region in the parameters space where

the parameters can move for a given solution without changing the objective values more than a predefined amount. The radius defines the degree of robustness: (i) if the radius is small, the solver will give us a solution specific to these parameters; (ii) if it is large, the solver gave us a more robust solution. In the first case (less robust solution), the solution will have better fitness value than in the second case.

Regarding the shortest path problem, there are several approaches to take into account the robustness. In [31], the authors use a minmax regret strategy to find a solution that minimizes the cost in the worst case scenario. Other authors as Cheng et al. [6] add a random delay, they optimize the travel times, in the optimization problem. This model of robustness allows adding some imprecision in the weight of the edge. The authors also describe how to use the knowledge of the probability distribution of this random variable. The authors solved this problem by applying a strategy similar to the minmax regret, using the supreme of the probability distribution of the delay. The main limitation of using minmax is that it only returns a single robust solution. In [20], the authors modeled robustness using confidence intervals. Although they tackled the single-objective problem, they transformed it into a multi-objective problem by moving from exact values to confidence intervals.

In spite of the fact that the BSP problem has been studied in very different variants [5, 29, 40], the RBSP problem has been less studied in the scientific literature. Some authors as Ehrgott et al. [11] or Kuhn et al. [27] have done works in robust optimization from a theoretical point of view.

Ehrgott et al. [11] classify robust solutions according to criteria of dominance between solutions. To solve the problem, they propose three approaches based on weighted sum scalarization, ϵ-constraint scalarization, and objective-wise worst case.

Kuhn et al. [27] present a classification of the different degrees of robustness of a single solution according to the number of scenarios in which it is efficient. However, they only take into account the uncertainty in one of the objectives, while we consider the two objectives with the same uncertainty criteria. They also present an algorithm based on the two-phase method to obtain the Pareto set of efficient solutions from the set of robust solutions that are not dominated by any other solution in the whole set of solutions.

In general, the state-of-the-art in the field of robustness is treated in very different ways. These approaches have in common that they offer a single robust solution. In contrast, our proposal gives a set of solutions that is calculated by computing the variance in all the objectives.

7. Conclusions

In this work, we present a new robust model to deal with the uncertainty in the parameters of a problem in smart cities: the bi-objective shortest path problem. We have proposed a new model for the robust bi-objective shortest path problem based on multi-objective optimization. We optimized simultaneously the mean and the variance of the costs of the routes. We have selected parameters of interest to citizens for the costs of the edges: travel times and gas emission (CO_2 and NO_x). To test our proposal, we used real data and different algorithms: Dijkstra, A*, Pulse, NSGA-II, and MOEA/D.

We measure the degree of robustness of each solution by its variance in the objective space. We check that, if we take into account the robustness, we can obtain more robust solutions at the expense of a very small penalty in the quality of the solutions.

We also check that not only the mean values of the objectives are opposed, but also the variances. This is an important result when we want to implement the robustness model in a final application. Not only it is interesting to choose a path with little variability, but also to make an order of preference in the objectives to optimize.

As future work, we will test our proposed model with other cities (e.g., New York City) and other problems (e.g., the urban waste collection [12]). In a different line of work, we want to analyze how to select the weights for solving the multi-objective problem using single-objecive algorithms and a weighted sum of objectives. We think that it is possible to use the graph to select the most appropriate weight combinations. Finally, our model is open to the addition of new objectives, like monetary cost of the routes (including fees of toll stations), other pollutants, noise, etc.

Acknowledgements

This research has been partially funded by the Spanish MINECO and FEDER projects TIN2014-57341-R, TIN2016-81766-REDT, and TIN2017-88213-R. C. Cintrano is supported by a FPI grant (BES-2015-074805) from Spanish MINECO. It has also been partially funded by the Universidad de Málaga, Andalucia TECH.

References

[1] Ardakani, M. K., & Tavana, M. (2015). A decremental approach with the A* algorithm for speeding-up the optimization process in dynamic shortest path problems. *Measurement: Journal of the International Measurement Confederation*, *60*, 299–307.

[2] Ben-Tal, A., El Ghaoui, L., & Nemirovski, A. (2009). *Robust optimization*. Princeton University Press.

[3] Chand, S., & Wagner, M. (2015). Evolutionary many-objective optimization: A quick-start guide. *Surveys in Operations Research and Management Science*, *20*, 35 – 42.

[4] Chassein, A. B., & Goerigk, M. (2015). A new bound for the midpoint solution in minmax regret optimization with an application to the robust shortest path problem. *European Journal of Operational Research*, *244*, 739–747.

[5] Chen, P. W., & Nie, Y. M. (2013). Bicriterion Shortest Path Problem with a General Nonadditive Cost. *Procedia - Social and Behavioral Sciences*, *80*, 553–575.

[6] Cheng, J., Lisser, A., & Letournel, M. (2013). Distributionally robust stochastic shortest path problem. *Electronic Notes in Discrete Mathematics*, *41*, 511–518.

[7] Cintrano, C., Chicano, F., & Alba, E. (2017). Robust Bi-objective Shortest Path Problem in Real Road Networks. In *International Conference on Smart Cities, Smart-CT 2017* (pp. 128–136). Springer, Cham. (Lecture no ed.).

[8] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *Trans. Evol. Comp*, *6*, 182–197.

[9] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*, 269–271.

[10] Duque, D., Lozano, L., & Medaglia, A. L. (2015). An exact method for the biobjective shortest path problem for large-scale road networks. *European Journal of Operational Research*, *242*, 788–797.

[11] Ehrgott, M., Ide, J., & Schöbel, A. (2014). Minmax robustness for multi-objective optimization problems. *European Journal of Operational Research*, *239*, 17–31.

[12] Ferrer, J., Alba, E., & Chicano, F. (2015). Sistema Inteligente para la Recogida de Residuos en las Ciudades basado en Predicciones de Llenado. In *I Congreso Ciudades Inteligentes* (pp. 319–324).

[13] Gabrel, V., Murat, C., & Thiele, A. (2014). Recent advances in robust optimization: An overview. *European Journal of Operational Research*, *235*, 471–483.

[14] Garaix, T., Artigues, C., Feillet, D., & Josselin, D. (2010). Vehicle routing problems with alternative paths: An application to on-demand transportation. *European Journal of Operational Research*, *204*, 62–75.

[15] Goberna, M., Jeyakumar, V., Li, G., & Vicente-Pérez, J. (2015). Robust solutions to multi-objective linear programs with uncertain data. *European Journal of Operational Research*, *242*, 730–743.

[16] wei Gong, D., na Qin, N., & yan Sun, X. (2011). Evolutionary algorithms for optimization problems with uncertainties and hybrid indices. *Information Sciences*, *181*, 4124 – 4138.

[17] Gong, D., Sun, J., & Ji, X. (2013). Evolutionary algorithms with preference polyhedron for interval multi-objective optimization problems. *Information Sciences*, *233*, 141 – 161.

[18] Gong, D., Sun, J., & Miao, Z. (2018). A set-based genetic algorithm for interval many-objective optimization problems. *IEEE Transactions on Evolutionary Computation*, *22*, 47–60.

[19] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, *4*, 100–107.

[20] Hasuike, T. (2013). Robust shortest path problem based on a confidence interval in fuzzy bicriteria decision making. *Information Sciences*, *221*, 520–533.

[21] Hausberger, S., Rexeis, M., Zallinger, M., & Luz, R. (2009). *Emission Factors from the Model PHEM for the HBEFA Version 3*. Technical Report I.

[22] Hazir, Ö., Erel, E., & Günalay, Y. (2011). Robust optimization models for the discrete time/cost trade-off problem. *International Journal of Production Economics*, *130*, 87–95.

[23] Ide, J., & Schöbel, A. (2016). Robustness for uncertain multi-objective optimization: a survey and analysis of different concepts. *OR Spectrum*, *38*, 235–271.

[24] Jin, Y., & Branke, J. (2005). Evolutionary Optimization in Uncertain Environments - A Survey. *IEEE Transactions on Evolutionary Computation*, *9*, 303–317.

[25] Keshavarz, E., & Khorram, E. (2009). A fuzzy shortest path with the highest reliability. *Journal of Computational and Applied Mathematics*, *230*, 204–212.

[26] Kjeldsen, T. H. (2001). John von neumann's conception of the minimax theorem: A journey through different mathematical contexts. *Archive for History of Exact Sciences*, *56*, 39–68.

[27] Kuhn, K., Raith, A., Schmidt, M., & Schöbel, A. (2016). Bi-objective robust optimisation. *European Journal of Operational Research*, *252*, 418–431.

[28] Li, K., Zhang, Q., Kwong, S., Li, M., & Wang, R. (2014). Stable matching-based selection in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, *18*, 909–923.

[29] Mickael Randour, Jean-François Raskin, O. S. (2015). Variations on the Stochastic Shortest Path Problem. In D. D'Souza, A. Lal, & K. G. Larsen (Eds.), *Verification, Model Checking, and Abstract Interpretation* (pp. 1–18). Berlin, Heidelberg: Springer Berlin Heidelberg volume 8931 of *Lecture Notes in Computer Science*.

[30] Mirjalili, S., Lewis, A., & Mostaghim, S. (2015). Confidence measure: A novel metric for robust meta-heuristic optimisation algorithms. *Information Sciences*, *317*, 114–142.

[31] Pascoal, M. M., & Resende, M. (2014). The minmax regret robust shortest path problem in a finite multi-scenario model. *Applied Mathematics and Computation*, *241*, 88–111.

[32] Pascoal, M. M., & Resende, M. (2015). Dynamic preprocessing for the minmax regret robust shortest path problem with finite multi-scenarios. *Discrete Optimization*, .

[33] Raith, A., Schmidt, M., Schöbel, A., & Thom, L. (2018). Multi-objective minmax robust combinatorial optimization with cardinality-constrained uncertainty. *European Journal of Operational Research*, *267*, 628–642.

[34] Riquelme, N., Von Lücken, C., & Baran, B. (2015). Performance metrics in multi-objective optimization. In *Computing Conference (CLEI), 2015 Latin American* (pp. 1–11). IEEE.

[35] Sedeño-Noda, A., & Raith, A. (2015). A Dijkstra-like method computing all extreme supported non-dominated solutions of the biobjective shortest path problem. *Computers & Operations Research*, *57*, 83–94.

[36] Serafini, P. (1987). Some considerations about computational complexity for multi objective combinatorial problems. In J. Jahn, & W. Krabs (Eds.), *Recent Advances and Historical Development of Vector Optimization: Proceedings of an International Conference on Vector Optimization Held at the Technical University of Darmstadt, FRG, August 4–7, 1986* (pp. 222–232). Berlin, Heidelberg: Springer Berlin Heidelberg.

[37] Sheng, Y., & Gao, Y. (2016). Shortest path problem of uncertain random network. *Computers and Industrial Engineering*, *99*, 97–105.

[38] Sun, J., Gong, D., Zeng, X., & Geng, N. (2018). An ensemble framework for assessing solutions of interval programming problems. *Information Sciences*, *436-437*, 146 – 161.

[39] Zhang, Q., & Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*, 712–731.

[40] Zhang, Y., Liu, P., Yang, L., & Gao, Y. (2015). A bi-objective model for uncertain multi-modal shortest path problems. *Journal of Uncertainty Analysis and Applications*, *3*, 8.