This is the published version of a paper published in *Advances in Operations Research*.

Citation for the original published paper (version of record):

Multiobjective Optimization for Multimode Transportation Problems
*Advances in Operations Research*
https://doi.org/10.1155/2018/8720643

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:
http://urn.kb.se/resolve?urn=urn:nbn:se:du-27779
Research Article
Multiobjective Optimization for Multimode Transportation Problems

Laurent Lemarchand 1, Damien Massé 1, Pascal Rebreyend 2, and Johan Håkansson 2

1 Lab-STICC UMR CNRS 6285, University of Brest, Brest, France
2 Dalarna University, Falun, Sweden

Correspondence should be addressed to Pascal Rebreyend; prb@du.se

Received 5 December 2017; Accepted 15 April 2018; Published 7 June 2018

Academic Editor: Alessandra Oppio

Copyright © 2018 Laurent Lemarchand et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We propose modelling for a facilities localization problem in the context of multimode transportation. The applicative goal is to locate service facilities such as schools or hospitals while optimizing the different transportation modes to these facilities. We formalize the School Problem and solve it first exactly using an adapted $\varepsilon$-constraint multiobjective method. Because of the size of the instances considered, we have also explored the use of heuristic methods based on evolutionary multiobjective frameworks, namely, NSGA2 and a modified version of PAES. Those methods are mixed with an original local search technique to provide better results. Numerical comparisons of solutions sets quality are made using the hypervolume metric. Based on the results for test-cases that can be solved exactly, efficient implementation for PAES and NSGA2 allows execution times comparison for large instances. Results show good performances for the heuristic approaches as compared to the exact algorithm for small test-cases. Approximate methods present a scalable behavior on largest problem instances. A master/slave parallelization scheme also helps to reduce execution times significantly for the modified PAES approach.

1. Introduction

Localization of facilities such as public services (schools, hospitals, etc.) is an important problem for social planners and policymakers. Most of the time, this problem is formulated as the (single objective) $p$-median problem which is a central problem in Operations Research (see, for example, [1, 2] for surveys). The $p$-median was introduced by Hakimi [3] who describes its basic properties. Its basic variant can be defined as follows: given a set of $N$ demand nodes, distance values for each pair of nodes, and a fixed number $p$ of facilities, locating each facility at one of the nodes, while minimizing the sum of distances from each node to its closest facility.

Recent developed algorithms solve the single objective version exactly for instances of thousands of nodes (e.g., 25,000 nodes in [4]). However, for a policymaker, considering additional objectives would be useful when solving $p$-median problems on real cases, leading to different variants; e.g.,

(i) dispersion problem: the $p$-dispersion problem consists of spanning the $p$ facilities by maximizing the minimal distance between two of them. This objective function is suitable to locate business franchises and also when locating obnoxious facilities [5].

(ii) $p$-center problem: the $p$-center problem [6] aims at minimizing the maximal distance of demand nodes from their facility, or the average distance of a fraction of those that are the farthest from their closest facility, e.g., 5% farthest of them. This problem formulation can be applied to locate emergency services such as fire stations.

(iii) multimode transportation location (MTL) problem: in many real cases, transportation can be done by different means (by foot, bike, car, buses, etc.) depending on criteria like a threshold on the distance to the nearest facility. As an example, pupils are going to school by
foot (category A) or by public transportation (category B), with a threshold on the distance defining the 2 categories, e.g., 2 km. The objective is then to minimize both the mean distance for those of category A and the number of people in category B.

In the following we will focus on the MTL problem with multiple objectives. The practical context is to optimize location of schools. This School Problem is a typical multimode transportation problem since, depending the distance, pupils can go to school by foot or by bus. MTL problems can be seen as multiobjective optimization problems if means of transportation have impact on each other. Optimizing the cost for one of them can degrade the other objective value.

When considering a multiple objective optimization problem (MOO), a single solution optimizing all of the objectives simultaneously rarely exists. Let \( f: S \rightarrow \mathbb{Z} \) be an objective function mapping solutions \( s \in S \), the search space, to the objective space \( \mathbb{Z} \), with \( \mathbb{Z} \subseteq \mathbb{R}^m \). MOO algorithms look for solutions \( s \in S \) such that \( z = f(s) \) is optimized (in the sequel, we consider minimization). \( z \in \mathbb{R}^m \) is a point of the objective space \( \mathbb{Z} \), with each of \( z_i = f_i(s), i \in [1..m] \) being one of the objective function values to be minimized.

Many approaches rely on the dominance concept to choose, among a set of solutions \( S \), those ones that represent the best trade-offs of objectives within the search space. We say that a solution \( s \in S \) dominates another one \( s' \in S \) if \( \forall i \in [1..m], f_i(s) \leq f_i(s') \) and \( \exists i \in [1..m], f_j(s) < f_j(s') \). It is denoted as \( s > s' \). Namely, \( s \) is as good as \( s' \) on all objectives and better than \( s' \) for at least one of them. Solutions that are not dominated by any member of \( S \) are efficient solutions and constitute the Pareto set \( \mathcal{P} \).

2. Related Work

A few works extend the \( p \)-median problem with multiple objectives. The \( p \)-median problem is \( NP \)-hard [12], and MOO versions are also \( NP \)-hard since they embed the single objective version. Thus, if some exact algorithms exist, heuristic approaches are preferred for large test-cases.

The MOO \( p \)-median problem with an additional facility cost objective is dealt with in [13]. Each facility is weighted by a building cost, and the goal is to minimize the sum of distances to locations (\( p \)-median objective) and the sum of costs of the opened locations. The authors in [13] used two approaches. The first is an \( \epsilon \)-constraint like formulation that is a mix of two-phases algorithms [14] and classicale-\( \epsilon \)-constraint approach [15]. According to the authors, it leads to a close approximation of the full Pareto front. Even if its second objective is different (facility cost instead of number of pupils by public transportation), the technique used is similar to our approach concerning the \( \epsilon \)-constraint problem formulation. However, since the number of nodes varies in our case for the distance count (for instance, pupils going to school by public transportation are not taken into account for the distance of demands from facilities), the method must be adapted, as it will be shown in Section 4.

The second approach used in [13] helps to handle large test-cases and is based on MOGA framework, with a path relinking local search procedure for mixing solutions. Problems with uniform demand at each location (unitary problems) and with up to 400 demands and 20 facilities are processed. As MOGA, NSGA2, which we are testing, also uses a population-based approach.

In [16], the authors formulate a biobjective \( p \)-median problem with the following objectives: (1) minimal distance to the closest facility (\( p \)-median traditional objective) and...
(2) maximization of the minimal distance between facilities (p-dispersion problem objective). They formulate and solve it as a MOIP, with an $\epsilon$-constraint algorithm and also with a customized approach, based on threshold fixing for the minimal distance between facilities. Taking into account the threshold leads to additional constraints in a single objective subproblem. The algorithm iterates on subproblems, in order to provide the exact Pareto front of the biobjective initial problem. Once again, our second objective is different and implies an adaptation of the $\epsilon$-constraint algorithm. Furthermore, the modified MOIP developed in [16] is specific to the dispersion problem, since it iterates on a threshold value of tsunami risk exposure, and the third one takes into account dispersion (minimal closeness of selected locations).

Problems similar to multiobjective $p$-median have to also be examined in the context of disasters management [17], or tsunami exposure. In [18], the authors solve approximately a 3-objective optimization problem for school localization. Similarly to our heuristic method, it uses the NSGA2 framework mixed with a local search procedure, but the objective is not exactly the same as with $p$-median problems, since the number of facilities is not fixed but driven by a cost minimization objective. The second objective is related to tsunami risk exposure, and the third one takes into account both the distance to school and the number of pupils not able to go to school, as it is considered too far away (meaning, to go by foot). As we will see in the next section, this objective is in fact a mix of ours.

The $p$-median problem and its multiobjectives variants have also some military applications. Localization of repairs, supply, or security facilities can be modeled and addressed as $p$-median problems (see [19] for a survey). Common objectives are cost of localization, coverage of areas, and rapidity of deployment and also security of the localization that must be resilient. As an example, in [20], a multiobjective model, solved using a genetic annealing heuristic, takes into account cost and response time for Canadian forces prepositioning.

3. The Multimode Transportation

Problem Modelling

We want to optimize the location of services, such as schools, in such a way that users go by foot to the closest service if it is possible (according to a threshold $\alpha$ on the distance they can walk) and others take public transports. In the following, we will focus on a particular case of multimode transportation problem, the School Problem. The School Problem is stated as follows: given a set of $D$ demand nodes holding pupils and a set of $N$ candidate nodes for locating schools, a value of $\alpha$, and distances $d_{ij}$ between each couple (demand node, candidate node) ($1 \leq i \leq D$, $1 \leq j \leq N$), define a set of $p$ locations for setting up schools (school nodes) among the $N$ candidate nodes with the following objectives:

(i) The mean distance of demand nodes to the closest school node is minimized for those demand nodes that have a distance to closest school node less than the threshold $\alpha$.
(ii) The total number of demand nodes with a distance to the closest school greater than $\alpha$ is minimized.

The modelling of the School Problem can be stated as follows:

\[
\begin{align*}
\text{minimize} & \quad \sum_{i=1}^{D} \left( h_i F_i + \sum_{j=1}^{N} d_{ij} Y_{ij} \right) / \sum_{i=1}^{D} h_i F_i \\
\text{minimize} & \quad \sum_{i=1}^{D} h_i (1 - F_i) \\
\text{subject to} & \quad \sum_{j=1}^{N} Y_{ij} = p \\
& \quad \forall i, 1 \leq i \leq D, \\
& \quad \forall j, 1 \leq j \leq N, \\
& \quad Y_{ij} = 1 \implies X_j = 1 \\
& \quad \forall i, 1 \leq i \leq D, \\
& \quad \sum_{j=1}^{N} Y_{ij} = 1 \\
& \quad \forall i, 1 \leq i \leq D, \\
& \quad \left( \sum_{\{j|d_{ij} \leq \alpha \}} X_j \right) = 0 \iff F_i = 0 \\
& \quad \forall i, 1 \leq i \leq D, \\
& \quad \forall j, 1 \leq j \leq N, \\
& \quad F_i \in \{0, 1\}, \\
& \quad X_j \in \{0, 1\}, \\
& \quad Y_{ij} \in \{0, 1\}
\end{align*}
\]

where

(i) $D$ is the number of demand nodes;
(ii) $N$ is the number of candidate nodes (number of possible locations for a school);
(iii) $p$ is the number of candidate nodes to be selected as school nodes;
(iv) $\alpha$ is the threshold on walking distance;
(v) $h_i$ is the number of pupils located at demand node $i$;
(vi) $d_{ij}$ is the distance between demand node $i$ and candidate node $j$;
(vii) $F_i$ is a decision variable, indicating if the pupils at node $i$ go to school by walk or not (category A nodes);
(viii) $X_j$ is a decision variable, indicating if the candidate node $j$ is selected or not as a school node;
(ix) $Y_{ij}$ is a decision variable, indicating if the demand node $i$ closest school node is $j$ or not. If $Y_{ij} = 1$, we say that (demand node) $i$ is covered by (school node) $j$.

Equations (1) and (2) reflect the objectives stated above, taking into account the number of pupils located at each demand node. Equation (3) fixes the number of selected school nodes. Equations (4) and (5) ensure that the single selected candidate node for pupils at a demand node is effectively a school node. Equation (6) ensures that pupils at a demand node are accounted to go walking if and only if a candidate node is selected for school location in the neighborhood of the node.

This formulation induces that candidate nodes are restricted to demand nodes, but it could be extended to an arbitrary set of candidate nodes. Also notice that distance data can correspond to Euclidean distances, with known geographical locations for the different nodes, or to shortest path values, computed with an algorithm as Floyd-Warshall if the nodes are embedded within a routing graph, with moving cost values on the edges (e.g., time by walking, distance, and transportation cost). In the latter case, the value and meaning of the threshold are to be adapted. The use of the threshold reflects the public Swedish policy where authorities offer free transportation to some pupils based on the walking distance to school. If $\forall 1 \leq i \leq D, h_i = 1$, we speak about the unitary version of the School Problem.

We show in the next section how to model this problem as a MOIP and how to solve it with an ad-hoc technique.

### 4. Exact Problem Formulation and Solving

It is possible to solve exactly multiple objective problems using $\varepsilon$-constraint approaches. These techniques require a linear formulation of the aimed problem. Since the first objective function of our model is nonlinear (1) and nonconvex, those techniques are not directly applicable. In Section 4.1, we present the problem as a multiobjective mixed integer linear program (MOMILP). Its resolution would allow for computing the optimal value of each objective (by removing other objectives from the formulation and using a MILP solver as CPLEX, [21]). However, even computing only the mean distance (our first objective) with this model required high execution times, as detailed in Section 4.1.

Therefore, we consider a simplified MOIP in Section 4.2 where the computation of the mean distance is replaced by a computation of the cumulative distance. Using this model, we can compute the Pareto front $\mathcal{PF}$ and associated solution set using an adapted $\varepsilon$-constraint [22]-like approach presented in Section 4.3. We show that this method provides the exact Pareto set for the problem stated in Section 3.

#### 4.1. MOMILP Modelling

We studied the translation of the School Problem to a MOMILP problem mainly to compute directly the minimum mean distance (1) using a MILP solver. Since this objective function is continuous, its linearization requires using continuous variables. Indeed, starting from the modelling of the School Problem presented in Section 3, we can linearize (1) using a new set of (continuous) variables $Z_{ij}$, which replaces the set of (Boolean) variables $Y_{ij}$. The semantics of $Z$ variables is the following: $Z_{ij} \neq 0$ if demand node $i$ is covered by school node $j$ and $i$ is in category $A$ (i.e., $F_i Y_{ij} = 1$) and, in this case, $Z_{ij} = 1 / \sum_{i=1}^{D} h_i F_i$.

Using this semantics, objective (1) becomes linear, as shown in (1a). Notice that, by removing the variables $Y_{ij}$, we “forget”, for each demand node $i$ in category $B$ (i.e., with $F_i = 0$), which school node covers it. It is possible to keep this information, but it is useless for the resolution of the problem (as long as there is at least one school).

To satisfy the semantics of the variables $Z_{ij}$ and specifically specify $1 / \sum_{i=1}^{D} h_i F_i$, we also add a continuous variable $G_i$ for each demand node $i$ and a continuous variable $R$. The resulting model is as follows:

$$\text{minimize } \sum_{i=1}^{D} \left( h_i \sum_{j=1}^{N} d_{ij} Z_{ij} \right)$$

subject to

$$\sum_{j=1}^{N} X_j = p$$

$$Z_{ij} \leq X_j \quad \forall i, 1 \leq i \leq D, \forall j, 1 \leq j \leq N$$

$$\sum_{j=1}^{N} Z_{ij} + (1 - F_i) \geq R \quad \forall i, 1 \leq i \leq D$$

$$F_i \geq X_j$$

$$\forall i, 1 \leq i \leq D, \forall j, 1 \leq j \leq N, d_{ij} \leq \alpha$$

$$F_i \leq \left( \sum_{[j(1) \leq j \leq N] d_{ij} = b} X_j \right) \quad \forall i, 1 \leq i \leq D$$

$$G_i \leq R \quad \forall i, 1 \leq i \leq D$$

$$G_i \leq F_i \quad \forall i, 1 \leq i \leq D$$

$$\sum_{i=1}^{D} h_i G_i \geq 1$$

$$X_j \in \{0,1\} \quad \forall j, 1 \leq j \leq N$$

$$F_i \in \{0,1\}$$

$$G_i \in \{0,1\}$$

$$\forall i, 1 \leq i \leq D$$

$$Z_{ij} \in \{0,1\}$$

$$\forall i, 1 \leq i \leq D, \forall j, 1 \leq j \leq N$$

$$R \in \{0,1\}$$

In this model, (R1), (R2) and (R3), ensure that $R \geq 1 / \sum_{i=1}^{D} h_i F_i$; from (R2) and (R3) (and since $F_i$ are binaries), we
have \( \sum_{i=1}^{D} h_i F_i G_i \geq 1 \). By (R1), we get \( \sum_{i=1}^{D} h_i F_i R \geq 1 \), which gives the lower bound of \( R \).

From this result, (5a) states that when demand node \( i \) is in category \( A \) (i.e., \( F_i = 1 \)), \( \sum_{j=1}^{N} Z_{ij} \geq R \) (the minimization of (1a) ensures that only one \( Z_{ij} \) is equal to \( R \) and the other \( Z_{ij} \)s are zero). Equation (4a) implements Constraint (4), whereas the implementation of Constraint (6) is done by (6a) and (6b) (note that the former is needed for the first objective while the latter is needed for the second one).

We have tried to exploit this formulation for computing the minimal mean distance, with the CPLEX MILP solver [21], using the single objective function (1a). In practice this approach did not work, except on our smallest example (100 nodes and 5 schools). On the other test-cases, CPLEX returned possibly nonoptimal solutions after reaching its memory limit. We think that this failure stems from two issues. The first issue is the variability of \( R \) (and, as a result, the values of all nonzero \( Z_{ij} \)) which makes node selection a hard problem. The second issue is that when solving relaxed problems, with \( F_i \) and \( X_i \) partly continuous, (5a) enables putting \( \sum_{j} Z_{ij} = 0 \) even when \( F_i \) is close to 1, since \( R \) is very small. As a result, the algorithm needs to force almost all \( X_j \) (and \( F_j \)) to integral values before getting a meaningful minimal bound. Since both issues are related to the computation of a mean distance, we considered a MOIP using the cumulative distance instead.

4.2. MOIP Modelling. The MOIP formulation of the School Problem with the cumulative distance can be seen as a simplification of the MOMILP formulation of the previous section. In fact, we just need to state that \( R = 1 \). As a result, the \( G_i \) variables are useless ((R1), (R2), and (R3) disappear). From this result, we propose the MOIP formulation \( \delta SP \) for the School Problem, as follows:

\[
\text{minimize } z_1 = \sum_{i=1}^{D} \sum_{j=1}^{N} h_i d_{ij} Z_{ij} \tag{9}
\]
\[
\text{minimize } z_2 = \sum_{i=1}^{D} h_i (1 - F_i) \tag{10}
\]
\[
\text{subject to } \sum_{j=1}^{N} X_j = p \tag{11}
\]
\[
Z_{ij} - X_j \leq 0 \quad \forall i, j, 1 \leq i \leq D, 1 \leq j \leq N \tag{12}
\]
\[
Z_{ij} = 0 \quad \forall i, j, 1 \leq i \leq D, 1 \leq j \leq N, d_{ij} > \alpha \tag{13}
\]
\[
\sum_{j=1}^{N} Z_{ij} = F_i \quad \forall i, 1 \leq i \leq D \tag{14}
\]
\[
F_i \geq X_j \quad \forall i, j, 1 \leq i \leq D, 1 \leq j \leq N, d_{ij} \leq \alpha \tag{15}
\]

Compared with the previous model, we replace (6b) by (12), (13), and (14), which ensure that demand node \( i \) cannot be in \( A \) if there is no school node at distance \( \alpha \).

Notice that (13) and (15) both use \( \alpha \), but with different meanings: the former ensures that a pupil cannot walk to a distant node (to minimize the second objective), while the latter prevents an artificial minimization of the cumulative distance by covering a demand by a distant node where a closer one is available.

As we have seen before, our School Problem is formalized only approximately by \( \delta SP \), since (9) uses the cumulative distance instead of the mean distance for pupils in category \( A \). While the minimization of the cumulative distance can be compared to the \( p \)-median problem (indeed, we can get the \( p \)-median problem by adding \( F_i = 1 \) for all \( i \) and removing (13)), its exact resolution using MOIP solvers can be more difficult, as the relaxed problem (with continuous variables) has more solutions. To explain this issue, let us consider only two demand nodes (which are also candidate nodes) 1 and 2 such that \( d_{12} \leq \alpha \) and one location (\( p = 1 \)). With the relaxed \( p \)-median problem (\( F_1 = F_2 = 1 \)), the solver gives directly the optimal cumulative distance \( d_{12} \), whatever the values of \( X_1 \) and \( X_2 \). Using the relaxation of \( \delta SP \), the optimal solution sets all variable to 1/2 except \( Z_{12} \) and \( Z_{21} \) which are set to 0 and the cumulative distance is 0.

Using this observation and the fact that the cumulative distance is still not the mean distance lead us to consider an adapted \( \varepsilon \)-constraint method where the first objective is (10).

4.3. Exact Algorithm. \( \varepsilon \)-constraint [22] methods proceed as follows: a series of single objective (i.e., Integer Program, IP) problems are solved, transforming all but one of the objectives of the MOIP considered into IP constraints. The constraint set is updated at each iteration to enforce the exploration of the whole objective space. In their paper [23], Ozlen and Azizo˘glu introduce a recursive algorithm to generate a Pareto set for a MOIP problem. They use the set of already solved subproblems and their solutions to avoid solving a large number of IPs. In [14], a two phases algorithm is applied for the biobjective assignment problem. In the 2 objectives case, it first determines the extreme points in the objective space by discarding one objective at a time and solving the resulting single objective problem. Then, in a second phase, it partitions the objective space according to the range of values found at the first step and explores it by slices. It also combines the approach with heuristics specific to the assignment problem for enhancing the execution times.

An adapted \( \varepsilon \)-constraint algorithm which computes all nondominated points sorted by \( z_2 \) values (i.e., second objective) is presented in Algorithm 1 for solving \( \delta SP \). This approach uses the fact that a given nondominated point \( z = (z_1, z_2) \), other nondominated point \( z' = (z'_1, z'_2) \) such that \( z'_2 \geq z_2 \) must satisfy both \( z'_1 < z_1 \) and \( z'_2 > z_2 \). Since the
minimal cumulative distance is hard to compute (and not really useful as it may not be the minimal mean distance), we do not compute the extreme point associated with it.

In $\mathcal{S}\mathcal{P}$, we consider the minimization of the cumulative distance instead of the mean distance. Hence both objective function parameters are of integer values (once all distances are converted into integers), which ensures that our $\varepsilon$-constraint method cannot miss any efficient solution for the cumulative distance problem. Furthermore, one can check that, for any feasible solution $s$ of value $z = (z_1, z_2)$, the mean distance for people in category $A$ is

$$\frac{z_1}{\sum_{i=1}^{D} h_i Z_{ij}} = \frac{z_1}{M - z_2}$$ (19)

where $M = \sum_{i=1}^{D} h_i$. Especially $M = D$ if $h_i = 1$ for all $i$ (Unitary Problem). Hence, any efficient solution for the mean distance problem is also an efficient solution of the cumulative distance problem.

**Theorem 1.** Let $s$ be a feasible solution for $\mathcal{S}\mathcal{P}$ (associated with point $z = (z_1, z_2)$ in objective space), such that it is efficient for mean distance optimization, i.e., for each feasible solution $s'$ (point $z'$):

$$\frac{z_1}{M - z_2} \leq \frac{z'_1}{M - z'_2}$$ (20)

or $z_2 \leq z'_2$

Then $s$ is efficient for cumulative distance optimization.

**Proof.** If $z_2 \leq z'_2$, the result is straightforward. Otherwise, $z_1/(M - z_2) \leq z'_1/(M - z'_2)$ and $M - z_2 < M - z'_2$, which implies $z_1 < z'_1$. \hfill $\square$

Thus all efficient solutions for the School Problem are found by solving $\mathcal{S}\mathcal{P}$. But the converse is not true, and nondominated points can also appear in the Pareto front for $\mathcal{S}\mathcal{P}$ that do not belong to the Pareto front for the School Problem. However, since the solutions generated are sorted by the value of $z_2$, it is easy to make the algorithm generate only nondominated points for the School Problem: for each generated nondominated point $z$, we add the following constraint to the problem:

$$\sum_{i=1}^{D} \sum_{j=1}^{N} ((M - z_2) d_{ij} - z_1) h_i Z_{ij} \leq -1$$ (21)

Constraint (21) (which has only integer coefficients) ensures that any subsequent solution has a mean distance strictly less than the previous one. Including this constraint, Algorithm 1 computes the Pareto front for the School Problem.

5. **Heuristic Problem Solving**

MOO problems search spaces are often intractably large [15]. Many heuristics have been developed for searching over huge spaces, particularly using Evolutionary Multiobjective Algorithms (EMOA). Genetic algorithms have been shown to be interesting for solving very large instances of the $p$-median problem in [24]. We investigate the extension of this work to the MOO case, focusing on chromosome-based EMOA. We investigate the use of two algorithms, NSGA2 and PAES.

NSGA2 is a reference algorithm in the EMOA field, and it compares positively to PAES for standard benchmark problems [8]. But as stated below, NSGA2 (and many other EMOAs) iterates onto a population of individuals (each of them represents a solution), producing eventually a large
offspring at each generation. Even if this is an advantage when exploring the search space for building an interesting front, it can be very costly when employing such an algorithm for solving a problem with a costly evaluation function. The evaluation function can be intrinsically complicated (e.g., physics simulation) or time costly because it runs a local search within the neighborhood of the solution to be evaluated. We have already applied PAES successfully for the former reason in the field of Real-Time Scheduling [25]. We have also used a local search technique for the single objective p-median problem in [24]. Thus our goal is to compare PAES coupled with a local search technique to the reference algorithm NSGA2 for the MOO School Problem.

We first present and compare shortly NSGA2 and PAES and then we detail our declination of both frameworks with a local search technique for solving the School Problem.

5.1. PAES and NSGA2. In many evolutionary algorithms, a solution is represented by a set of parameters called chromosome (or genotype). The encoding of solutions (i.e., the data structure of a chromosome) is specific for each problem. Several researchers have proposed genetic algorithms (GA) for solving the p-median problem [26, 27]. Most of them use a classical string representation; i.e., each chromosome is represented by a single array of integers of length p embedding the index of the selected candidate nodes. As stated in [24], we will use the same encoding, i.e., the chromosome represents the list of school nodes in our case. We add the constraint that in all chromosomes no school is duplicated. The initial population (or first solution) of our algorithms is generated by picking up distinct random candidate nodes, leading to feasible solutions. All the candidate nodes have the same probability to be chosen.

Both EMOA’s goals are to produce a front that preserves diversity of solution objective values, with points that are numerous and spread well along PF and the accuracy of this front, with points that are close to PF. Within a population set of solutions, both PAES and NSGA2 use a crowding metric for handling diversity: PAES by defining an adaptive grid over the objective space and counting points within each portion of the grid and NSGA2 by measuring the distance between points and their closest neighbor in each direction for each objective. To ensure accuracy, both algorithms embed an elitism mechanism: the number of solutions that are kept across generations (elitism) is given either by the population size (NSGA2) or by an external archive (PAES). This size also controls the number aspect of diversity. In order to increase accuracy of solutions and convergence to PF, PAES uses dominance locally, comparing current solution to a single offspring at each generation, while NSGA2 sorts its population into dominance classes. PAES eventually updates its archive with the offspring, while NSGA2 renews its population by generating a series of offsprings at each generation, with a dominance class based selection. With both algorithms, when comparing solutions, if they are nondominated against each other, the crowding metric is used to tie them. Concerning the operators used for generating offspring, we have implemented a specific mutation operator that preserves feasibility within PAES and NSGA2 applies standard binary mutation and crossover onto a binary representation of chromosomes it generates internally. We also provide a penalty metric that helps NSGA2 discarding unfeasible solutions (penalizing duplicated schools).

We will see in the two next sections how to adapt those strategies for mixing the various approaches with a fast local search technique inspired from p-median single objective literature.

5.2. Local Search Technique for MOO. Many heuristic methods have been developed for the (single objective) p-median problem (see surveys [1, 2]). Variable neighborhood search [9] is very popular, coupled with fast evaluation techniques [1]. Those techniques, based on precomputing of the 2 closest schools for each demand node, allow updating the solution cost faster when modifying only partially the solution itself during the local search process: for each demand node, for a given set of schools (solution), the distance to closest school ($s_1$, the one that covers the demand node) and the second closest school ($s_2$) and its distance to the node are stored. We use a neighborhood operator called hypermutation, inspired from [24]. The neighborhood size is controlled by the number of modified school nodes (seeds): the neighborhood of a solution, for a fixed number of seeds $\tau$, is defined by replacing one by one each seed node by all of the possible other candidate nodes; i.e., those which are not yet selected as school nodes within the solution. The solution cost can be updated faster using the precomputed tables mentioned above: when a school $s$ is replaced by $s'$, demand nodes covered by it ($s = s_1$) are now covered by their second closest school ($s_2$), except if $s'$ is closer to them than $s_2$. Objective functions are updated according to the case. We fix the number of seeds to $\tau = \min(p,5)$ for a School Problem, to obtain a reasonable neighborhood size. This size is $\tau(n-p)$, with $\tau$ the number of seeds, $n$ the number of candidate nodes, and $p$ the number of schools.

The neighborhood and associated evaluation techniques are devoted to the single objective p-median problem. They can be kept when running the MOO case. However, evaluating and comparing to current front multiple neighbors resulting from multiple function evaluations could be very time consuming: evaluation itself can be costly if applied to a large number of neighbors, and these numerous solutions must be compared against existing population or archive, with a dominance checking procedure. Thus, we propose an approach where the neighborhood exploration is run with a single objective metric and only the most interesting solutions found during the local search are compared to the current solutions kept in population or archive. This algorithm is outlined in Algorithm 2. We transform, for the local exploration, the 2 objectives metric of the School Problem into a single one by a classical aggregation technique: weights are given to objectives and $f_{\text{single}} = \sigma.z_1 + (1-\sigma).z_2$ is computed. $z_1$ and $z_2$ are normalized values (according to the objective vector of the neighborhood search starting point). $f_{\text{single}}$ is evaluated for different values of $\sigma$ in order to explore the objective space into different directions (e.g., $\{0.0, 0.25, 0.50, 0.75, 1.0\}$). Since $\sigma$ is restricted to a few values, only a subset of the supported solutions can be found and of course unsupported ones
Data: a solution sol (set of p school nodes) to be evaluated, n, p (number of candidates nodes and schools), a set dirs of λ directions, a number of seeds τ
Result: as offspring of λ solutions
begin
foreach σ ∈ dirs do
  best[σ] := (−, ∞) { (solution, value) couples }
end
(z1, z2) := evaluate(sol)
choose randomly a subset seeds ⊂ sol of size |seeds| = λ
foreach school s ∈ seeds do
  foreach neighbor solʹ = sol ∪ { s′ ∈ sol } \ { s } do
    (z1′, z2′) := fastEvaluate(solʹ)
    { based on sol and (z1, z2) }
    foreach σ ∈ dirs do
      V := z1′.σ + z2′.(1 − σ)
      if V < value(best[σ]) then
        best[σ] := (solʹ, V)
      end
    end
  end
end
offsprings := 0
foreach σ ∈ dirs do
  offsprings := offsprings ∪ { solution(best[σ]) }
end
end

Algorithm 2: A local search method for supported solutions in λ directions.

cannot be detected by the local search [28]. For each direction searched, only the best solution (i.e., with the minimal \( f_{single} \) value for the associated \( σ \) weight) into the neighborhood is provided as a result at the end of the local exploration process. This approach can be related to the decomposition techniques used in algorithms like MOEA/D [29].

5.2.1. Cost of Evaluation. The grain of the search is tuned by the number of seeds \( τ \), leading to a neighborhood of size \( τ.(n − p) \), and also by the number \( λ \) of different values for \( σ \). Let \( T \) by the cost of a (standard) evaluation of a solution. It mainly consists of finding the closest school for each children and is in \( O(\lambda.n^2.p) \). Evaluating a series of neighbors for a neighborhood of \( (r.(n − p)) \) costs \( O(\tau.(n − p)n^2.p) \), also stated as \( T_n = O(\tau.(n − p).T) \). During the (fast) evaluation of a solution as a neighbor of another one, only a single school is modified and only the distances to school for its covered nodes are to be updated, in \( O(n/p) \) time. Thus evaluating a whole neighborhood takes \( T_n = O(\tau + \lambda.\tau.(n − p)n/p) \), with roughly, a \( n.p^2 \) factor of gain as compared to \( T_n \).

5.3. Mixing the Local Search Technique with EMOA Algorithms. This local exploration procedure can be mixed with the different EMOAs into 2 different ways:

(i) Applying the local search for each individual: the offspring of an individual is generated via the local search procedure, with λ directions of search.

(ii) Using the local search as a refinement step: applying local search to the members of the population or archive obtained as a postprocessing step.

NSGA2 processes natively multiple offsprings, but the original PAES manipulates a single current solution and a single offspring (it is a \((1 + 1)\) procedure) that replaces (or not) the current solution at next generation, depending on dominance checking. In the first way of mixing, a \((1 + 1)\) algorithm such as PAES must be adapted. Knowles proposes such a \((1 + \lambda)\) version in [7]. We use instead the same selection mechanism as in [30], developed for costly functions coupling with PAES and well adapted in the context of an hybrid method combining EMOA with (time costly) local search. The selection procedure is as follows: the \( λ \) offsprings resulting from the local search are compared to the PAES archive using PAES policy. Then, the new current solution is chosen considering the whole archive as follows, based on a randomly selected optimization criteria: a random integer \( r \) in the interval \([1..3]\) is generated. If it corresponds to an objective function (i.e., \( r \in [1..2] \)), these criteria are chosen;
Table 1: Number of candidate nodes $n$ and school nodes $p$ for the 40 test-cases of the Beasley benchmark.

<table>
<thead>
<tr>
<th>#</th>
<th>n</th>
<th>p</th>
<th>#</th>
<th>n</th>
<th>p</th>
<th>#</th>
<th>n</th>
<th>p</th>
<th>#</th>
<th>n</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>5</td>
<td>11</td>
<td>500</td>
<td>5</td>
<td>21</td>
<td>500</td>
<td>5</td>
<td>31</td>
<td>700</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>10</td>
<td>12</td>
<td>300</td>
<td>10</td>
<td>22</td>
<td>500</td>
<td>10</td>
<td>32</td>
<td>700</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>10</td>
<td>13</td>
<td>300</td>
<td>30</td>
<td>23</td>
<td>500</td>
<td>50</td>
<td>33</td>
<td>700</td>
<td>70</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>20</td>
<td>14</td>
<td>300</td>
<td>60</td>
<td>24</td>
<td>500</td>
<td>100</td>
<td>34</td>
<td>700</td>
<td>140</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>33</td>
<td>15</td>
<td>300</td>
<td>100</td>
<td>25</td>
<td>500</td>
<td>167</td>
<td>35</td>
<td>800</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>200</td>
<td>5</td>
<td>16</td>
<td>400</td>
<td>5</td>
<td>26</td>
<td>600</td>
<td>5</td>
<td>36</td>
<td>800</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>200</td>
<td>10</td>
<td>17</td>
<td>400</td>
<td>10</td>
<td>27</td>
<td>600</td>
<td>10</td>
<td>37</td>
<td>800</td>
<td>80</td>
</tr>
<tr>
<td>8</td>
<td>200</td>
<td>20</td>
<td>18</td>
<td>400</td>
<td>40</td>
<td>28</td>
<td>600</td>
<td>60</td>
<td>38</td>
<td>900</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>200</td>
<td>40</td>
<td>19</td>
<td>400</td>
<td>80</td>
<td>29</td>
<td>600</td>
<td>120</td>
<td>39</td>
<td>900</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>200</td>
<td>67</td>
<td>20</td>
<td>400</td>
<td>133</td>
<td>30</td>
<td>600</td>
<td>200</td>
<td>40</td>
<td>900</td>
<td>90</td>
</tr>
</tbody>
</table>

otherwise ($r = 3$) the crowding criteria is selected. The next current individual is chosen randomly within the 10% best ones within the archive for the elected criteria. Concerning crowding, the best individuals are those in the less crowded areas of the grid.

The goal of the approach is to solve large size problems with a high execution time cost induced by evaluation and proportionally a small amount of time devoted to EMOA internal computations. So we consider here that the cost of choosing next current individual by processing the whole archive is not a drawback for execution times according to the benefit in terms of quality of search that we expect.

6. Algorithms Comparison

The multimode transportation problem or School Problem to be solved is specific, and to our knowledge, no comparable algorithm exists for solving it (see Section 2). Thus, this evaluation section mainly compares the heuristic approaches that we have developed to our customized $\epsilon$-constraint method. The following algorithms are compared:

(i) Exact: the $\epsilon$-constraint method presented in Section 4, which provides the exact Pareto Front $\mathcal{P}_F$,

(ii) EMOA approaches presented in Section 5, using NSGA2 and PAES algorithms. For both EMOAs, the standard and the hybrid versions tested: local search realized at each generation with PAES and as a postprocessing step for NSGA2.

We first describe the benchmark set used for comparing algorithms results, detail algorithms setup, and then present results for both quality and execution time aspects.

(a) Data Set. The data set used for the evaluation is the Beasley benchmark set, devoted to the $p$-median problem [11] and widely used for testing $p$-median solvers [26, 31, 32]. It is a set of 40 test-cases. Candidate and demand nodes are identical, with a unitary demand at each node. $n$ (the number of nodes) varies from 100 to 900 and $p$ value from 5 to 200. Testcases are ordered by increasing $n$ value and, for a given $n$, by increasing $p$ value as shown in Table 1.

We compute the $\alpha$ threshold in such a way that 15% of the distances between nodes are lower than $\alpha$.

The largest graph of this benchmark has 900 candidate nodes and is not particularly large according to the size of problems current $p$-median solvers can manage (e.g., instances of up to 24,978 nodes solved exactly in [4]). However, multiple objective optimization could lead to much more costly evaluation of the solution space for the same instances as compared to single-objective formulations, and Beasley benchmark contains large enough instances for our purpose, as execution times will show. It is also comparable to other biobjective $p$-median problem instances used with others approaches (e.g., 402 nodes for the largest instance in [13]).

(b) Algorithms Setup. The exact method has been implemented in C. It is a modified version of the AIRA software, a general purpose MOIP solver [33], and it uses CPLEX solver 12.3 as IP solver software [21]. As heuristic, we use the version of NSGA2 provided by Deb at [34], with the following parameters: a population size of 100, and 500 generations. We also use our own implementation of PAES, based on the C version provided by J. Knowles at [35]. The depth parameter of the algorithm is used to define the grain of the grid used for measuring the crowding of the objective space by the members of the PAES archive. It defines the number of recursive subdivisions of the range of values for each objective. It is set to 4, according to PAES recommendations for some biobjective problems. The archive size for PAES is of 100 and the number of generations of 1000. The parameters for the local search are $\lambda = 5$ (number of directions for search), corresponding to $\sigma \in \{0.00, 0.25, 0.50, 0.75, 1.00\}$ and $\tau = \min(p, 5) = 5$ for the number of seeds, since $p$ is always greater than 5 for the Beasley benchmark test-cases (see Table 1). Those parameters have been set in order to equilibrate the computational effort of the different algorithms, as execution times comparison will show. They lead to 1000 (resp. 50.000) standard evaluations and 25.000 fast evaluations for hybrid PAES (resp. hybrid NSGA2) (see Section 5.2). All of the tests are performed onto a 48 processors (Xeon E5 at 2.2GHz) SMP machine, with 132GB of memory, and running Linux CentOS.

(c) Quality Evaluation Procedure. The sets of solutions provided by the algorithms are to be compared qualitatively. Many metrics exist [28] that must take into account both the quality of solutions obtained (accuracy) and their number and location along the Pareto front range (diversity). The hypervolume (HV) [10] is often used to compare 2 sets of solutions $f_{\mathcal{F}_1}$ and $f_{\mathcal{F}_2}$. It computes the area defined by each set; according to a reference point dominated by all of the points in the sets, the highest value is the best one. Figure 1 illustrates the comparison by HV for 2 fronts, with two objective functions $f_1$ and $f_2$ to be minimized.

For each test-case, we run each algorithm in competition 30 times. We collect all of the resulting individuals and compute the associated non-dominated fronts. This front’s bounds on objectives are then computed and used for normalizing the objective vectors associated with the individuals into the
range [1, 2]. The point (2.1, 2.1) is thus dominated by all of the vectors and can be used for computing hypervolumes (see values in Figure 1), leading to a maximal possible HV of \((1.1)^2 = 1.21\), if a single objective vector dominates all of the others. We provide results for each algorithm and also make a statistical analysis with the Kruskal-Wallis nonparametric test [37], for comparing the quality value series obtained by running each algorithm 30 times for a given test-case: if and only if a first test for significance of any differences between the samples is passed (H0 hypothesis), at a given confidence value (we use the standard 0.05 value), then the output will be the one-tailored p values for rejecting a null hypothesis of no significant difference between two samples, for each pairwise combination. If the p value for a test-case is less than 0.05, we considerate that one algorithm beats the other for this test-case with a sufficient confidence. On the contrary, if series are not different enough, or the p value obtained for characterizing the differences is over 0.05, the test-case is not taken into account for comparing the couple of algorithms.

6.1. Hypervolumes Comparison. We compare the hypervolumes obtained for 5 algorithms:

(i) the Exact algorithm described in Section 4. The 10 first test-cases of the Beasley benchmark can be solved in less than 1 hour with this method, providing a reference front for comparison with heuristics for this subset of the Beasley benchmark.

(ii) the PAES algorithm, with our selection method, but without local search.

(iii) the hybrid PAES algorithm, embedding local search at each generation.

(iv) the NSGA2 algorithm,

(v) and the hybrid NSGA2 algorithm, using the local search as a postoptimization method

All the heuristics are applied onto the whole benchmark, with parameters as previously described. We have removed from the resulting mean values 5 test-cases (#20, #24, #25, #30, and #34): for those ones, NSGA2 (and thus also hybrid NSGA2) does not provide any feasible solution for at least 5 runs (all runs for #30). This is due to the fact that unfeasibility corresponds to duplicated schools in solutions, and this can arise with a higher probability when \(p\) increases. For the discarded test-cases, \(p\) is between 100 and 200 (see Table 1). Statistics are thus calculated over the 35 remaining test-cases.

Figure 2 shows the average values of hypervolumes for the different algorithms in competition. On average over the runs for the 40 test-cases, hybrid PAES provides the best results for all test-cases, when considering only EMOAs. It outperforms, respectively, PAES of 24.9%, NSGA2 of 58.7%, and hybrid NSGA2 of 48.0% on average. For the 10 test-cases for which exact solution is known, it provides a front with an HV that is, on average, at 7% of the optimal one. Clearly, the generic NSGA2 framework do not allow to obtain good results, as compared to the specialization of PAES for the School Problem: within NSGA2 version, the only specific component is the objective function, with nonspecialized mutation and crossover operators that can lead to unfeasible solutions. This is managed with NSGA2 by constraints penalties, but, unfortunately, this mechanism does not allow an efficient search of the feasible solution space. As a symptom, the size of the fronts provided by the NSGA2 algorithm is on average 68% less that those found by hybrid PAES (63% less for hybrid NSGA2). It is 49% when comparing PAES and hybrid PAES, showing that local search effectively helps in discovering nondominated solutions. The result is that local search allows improving results: hybrid PAES outperforms PAES by 24.9% and the local search by hybrid NSGA2 allows improving NSGA2 results in terms of HV by 32.7% on average. The latter is very significant, for a single local search phase, but the improvement is realized on the (relatively to hybrid PAES) poor results obtained with NSGA2, which leads room for optimization.

![Figure 1: Hypervolume (here an area, for 2 objective functions) for 2 fronts (inspired from [36]). Objective functions are to be minimized.](image1)

![Figure 2: Average compared quality of the solution obtained by the different algorithms for 30 runs on the 40 Beasley benchmark test-cases (10 first ones for the Exact method). Quality of a solution is expressed as its hypervolume.](image2)
Table 2: Number of test-cases, over the 40 test-cases of the Beasley benchmark, for which the Kruskal-Wallis nonparametric test validates the hypothesis that EMOAs beat each other for hypervolume.

<table>
<thead>
<tr>
<th>beats</th>
<th>PAES</th>
<th>hybrid PAES</th>
<th>NSGA2</th>
<th>hybrid NSGA2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAES</td>
<td>-</td>
<td>0</td>
<td>35</td>
<td>32</td>
</tr>
<tr>
<td>hybrid PAES</td>
<td>39</td>
<td>-</td>
<td>38</td>
<td>36</td>
</tr>
<tr>
<td>NSGA2</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>hybrid NSGA2</td>
<td>7</td>
<td>1</td>
<td>39</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 3: Compared mean execution times (30 runs) for different algorithms on the Beasley benchmark. Exact algorithms execution times, not presented, range from hours.

(a) Parallelization. Hybrid PAES is promising in terms of quality as compared to (hybrid) NSGA2 but time consuming as compared to PAES. We have implemented a master-slave scheme for realizing the evaluation and local search in parallel for different individuals, inspired from [30]. Parallel hybrid PAES implementation (in C) uses multiple threads (one per slave plus one for the master process). The OS scheduler allocates each of them to a core if the number of cores is sufficient. Figure 4 shows the speedups

\[
\frac{\text{exec. time parallel hybrid PAES with 1 slave}}{\text{exec. time parallel hybrid PAES with X slaves}} = \tau(n - p) / (n - p + \tau)
\]

obtained when running the algorithm with more than one slave (and thus more than 2 cores) for the evaluation and local search of individuals. The execution times are obtained using a 48 cores SMP system, ensuring that each slave (and the master) is run on a separate core. In order to increase the computational effort, the number of generations is set to 5000 instead of 1000 in previous tests.

Clearly, adding computing resources helps in speeding up the computations and especially when the computational effort is high (last test-cases). The average speedup is 1.89 (resp. 3.76, 7.48, 14.07, and 18.10) for 2 (resp. 4, 8, 16, and 32) slaves. Speedups look linear according to the number of slaves for up to 8 slaves. Classically, the cost of the parallelization is due to the bottleneck in communications induced by the master-slave scheme and also to parallelization itself: for example, for test-case #35, the sequential version of hybrid PAES runs in 29 seconds for 1000 runs (see Figure 3) and the parallel version, with a single slave, runs in 273 seconds for 5000 generations (instead of expected 29 seconds × 5 = 145 seconds). The communication bottleneck degrades the speedup for 16 slaves, but execution times are still improved in all cases as compared to 8 slaves version, except for test-case #5 (average speedup of 14.07). But for 32 slaves, the improvement of speedup is very limited (18.10 on average). As we find again the same steps shape on curves as those observed in Figure 3 for sequential times, with roughly higher speedups for the most costly test-cases, it is possible that the
A parallel version of hybrid PAES is also proposed, using a heuristic method called hybrid PAES mixes the 10 smallest instances of the Beasley public benchmark. A approach solves in a few minutes to one hour each of the Integer Programming formulation, is defined. The IP-based is proposed, and an exact resolution method, based onto an walking solution. A modelling for the School Problem namely, by foot and by public transport, with constraints on transportation mode for pupils, with two possible modes, localization problem, applied to a multimode transportation We present in this paper a MOO modelling of a facilities setup for the experiments and that it will not hold for larger instances. We plan to apply it to real data, as realized in [24] for single objective p-median problems, dealing with country-scale instances, with nonuniform population distribution. This is important because real problems may exhibit particularities in their data. Another direction of research is to add a third objective function, related to facility implantation costs, with a relaxation of the number of facilities (p becomes a bound instead of a fixed value). This economic cost is useful, for example, in disasters management.

**Conflicts of Interest**

The authors declare that there is are conflicts of interest regarding the publication of this article.

**References**


