A Novel Maximum Dispersion Territory Design Model Arising in the Implementation of the WEEE-Directive

Elena Fernández
Department of Statistic and Operations Research
Universitat Politècnica de Catalunya, Spain

Jörg Kalcsics
Chair of Operations Research and Logistics
Universität des Saarlandes, Germany

Stefan Nickel
Chair of Operations Research and Logistics
Universität des Saarlandes, Germany

Roger Z. Ríos-Mercado
Graduate Program in Systems Engineering
Universidad Autónoma de Nuevo León, Mexico

March 13, 2009

Abstract

The problem discussed in this paper is motivated by the new recycling directive WEEE of the EC. The core of this law is, that each company which sells electrical or electronic equipment in a European country has the obligation to recollect and recycle an amount of returned items which is proportional to its market share. To assign collection stations to companies, in Germany for one product type a territory design approach is planned. However, in contrast to classical territory design, the territories should be geographically as dispersed as possible to avoid that a company, respectively its logistics provider responsible for the recollection, gains a monopoly in some region.

First, we identify an appropriate measure for the dispersion of a territory. Afterwards, we present a first mathematical programming model for this new problem as well as a solution method based on the GRASP methodology. Extensive computational results illustrate the suitability of the model and assess the effectiveness of the heuristic.

Keywords: Heuristics, Optimization, Logistics, Recycling, Territory Design

Introduction

In 2003, the new recycling directive WEEE of the EC came into force, see (WEEE, 2003). WEEE stands for: waste electrical and electronic equipment. The core of this law is, that each company which sells electrical and electronic equipment in a European country has the obligation to recollect and recycle an amount of returned items which is proportional to the market share of the company in that country, and that this should be made at no cost for the last owners. To facilitate that approach, regional collection stations are set-up where the inhabitants can return their items free of cost at the end of their lifetime. If a collection device, usually a container or iron-barred box, at a station is full, one of the companies has to pick up the device and provide in exchange an empty device. In the directive, electric and electronic equipment is classified into ten categories. In 2005, this directive was transferred into German and Spanish law. Even if the law allows the individual collection and recycling of WEEE generated by each producer or distributor, it favors that companies engage in a common corporation (also called collection scheme) to jointly collect an amount of WEEE equal to the sum of their corresponding market shares. In Germany and Spain, the coordination and supervision of the collection is done by a central registry which also determines the market share of all companies that sell equipment in the respective country.

In the German recycling system, collection stations (which are operated by the local municipalities) are currently assigned dynamically to corporations. That is, whenever a collection device is full, the corporation that should perform the waste treatment is determined at short notice (based on the market share of the corporation, the overall amount of returned items, and the amount it has collected so far) and has to pick up the collection device within 48 hours. This system currently covers all product categories. However, for one group of products, the so-called white goods (i.e., dish-washers, dry-cleaners, washing machines, fridges, etc.), a territory design is in preparation. Here, each collection station should be statically assigned to a specific corporation for a given period of time. Whenever a collection device is full, always the same corporation is responsible for the waste treatment during that time. Essentially, the assignment of stations to corporations should be made such that the average amount of returned items is proportional to the market share of the corporation. Moreover, the stations assigned to a corporation should be evenly dispersed all over Germany to avoid a monopolistic concentration. Although this is from a logistics point of view not a satisfactory solution, another reasoning was predominant. Namely, before the implementation of the law a lot of smaller regional institutions and logistics providers were involved in the re-collection and recycling process. The legal authorities feared that these entities would be pushed out of the market if the territories were to be designed in the classical sense. Because in that way, corporations could minimize their administrative overhead by sub-contracting only a few larger providers that cover their whole territory. With dispersed territories this is (more) unlikely. Note that this criterion is exactly the opposite of the usual compactness and contiguity criteria of classical territory design problems, see (Kalcsics et al., 2005).

In this paper we will focus on the territory design collection system planned for the white goods in Germany. To the best of our knowledge, this specific problem has not yet been studied. Although a closely related problem has been addressed recently by (Grunow and Gobbi, 2008), which study the problem of assigning collection stations in Denmark to collection schemes which perform the waste treatment for the producers, some of their assignment criteria differ considerably from our problem. Within the context of the WEEE and its implementation, (Queiruga et al., 2008) discuss the location of recycling plants in Spain and present a PROMETHEE based approach to provide a selection of good alternatives for potential locations. Recycling systems found in other European countries are, e.g., described in (Hischier et al., 2005) (Switzerland) and (Turner and Callaghan, 2007) (UK). For a general overview on territory design models and an introduction into the topic the reader is referred to (Kalcsics et al., 2005), (Kalcsics, 2006), and references therein.

The remainder of the paper is organized as follows. In the next section we give a formal description of the problem, discuss appropriate measures for the dispersion of territories, and present a mathematical model. Afterwards, we introduce a GRASP framework for solving the problem. We present three different heuristics for constructing initial solutions as well as neighborhoods for the local search phase. In the subsequent section we give extensive computational results illustrating the suitability of the heuristic. The paper concludes with a summary and an outlook to further research.

Problem Description and Mathematical Model

In this section we will first describe the basic components, restrictions, and goals of the problem before discussing different dispersion measures and the mathematical model.

White goods are further subdivided into devices that have freezing capabilities and those that do not (denoted products of type 1 and type 2, respectively). This distinction is due to the toxic cooling solvents contained in the former products that require a special treatment. It is expected that for reasons of efficiency the 250 companies that currently distribute white goods in Germany will join into a few larger corporations that will arrange and carry out the recollection and recycling of the WEEE. The market share of each corporation is the sum of the market shares of its companies and is given for both product types separately.

To facilitate the territory design approach, Germany is divided into approximately 450 basic areas (the division is mainly based on existing administrative districts, like cities or counties). As there are no reliable estimates on the number of returned devices yet, the average amount of WEEE of a basic area is assumed to be proportional to the number of households. Moreover, based on the logistic effort to perform the recollection and recycling (measured by means of suitable performance indicators), all basic areas are classified into three groups: good, mediocre, and bad. A good basic area is for example one, which has a relatively small geographic extent, contains few collection stations, and possesses a good infrastructure. The motivation for this classification is that the actual costs for the recollection and recycling should also be (more or less) proportional to the market shares of the corporations. As the market shares may differ for the two product types, it is allowed to split basic areas, i.e., for some basic areas the corporation that collects products of type 1 is not the same as the one that is responsible for the type 2 products.

The task is now to assign basic areas fully or partially to corporations such that

1. for both product types all basic areas are assigned to a corporation;
2. for each corporation, the total number of households of all basic areas assigned to the corporation is proportional to its market share for each of the two product types;
3. the good, mediocre, and bad basic areas are evenly distributed among the corporations relative to their market shares;
4. the number of split basic areas is not too large;
5. for each corporation, all basic areas that are fully or partially assigned to the corporation are as dispersed as possible.

The set of all basic areas assigned to a corporation for at least one of the two product types is called a territory. The current solution method intended by the central registry assigns basic areas to corporations in a greedy-like fashion such that number of households is proportional to the market share of each corporation. If and how the other criteria are taken into account is not known.

Remark. Although not explicitly considered so far in the planning, an interesting additional criterion would be to look at the "value" of the waste returned, as this value may vary among regions due to different "habits" of wear and tear. Analogously to the logistics indices we could determine for each basic area a quality index and then also consider this criterion for the distribution of basic areas among corporations.

Before discussing appropriate measures for the dispersion of territories, we introduce some notation. Denote $V = \{1, \ldots, n\}$ the set of basic areas. Let w_i be the number of households of basic area $i \in V$ and $W = \sum_{i \in V} w_i$. Denote by V_1, V_2, and V_3 the set of good, mediocre, and bad basic areas. Use $q \in Q = \{1, 2, 3\}$ as an index for the respective sets and denote $q_i \in Q$ the logistics index of basic area i. Let d_{ij} be the distance between basic areas i and j, $i, j \in V$. We denote $C = \{1, \ldots, m\}$ the set of corporations and MS_k^p the market share of corporation $k \in C$ for product $p = 1, 2$. A solution is represented by a collection $X = \{X_k\}_{k \in C}$ with $X_k \subset V$. X_k represents the subset of basic areas that define the territory of corporation k and $X_k = X_k^1 \cup X_k^2$, where X_k^p denotes the subset of basic areas assigned to k for product $p = 1, 2$. If basic area $i \in V$ is non-split we have $i \in X_k^1 \cap X_k^2$, for some k; otherwise, there exist k_1, k_2, $k_1 \neq k_2$, with $i \in X_{k_1}^1 \cap X_{k_2}^2$. When no splitting is allowed, we have $X_k = X_k^1 = X_k^2$, for all $k \in C$, so that $X = \{X_k\}_{k \in C}$ defines a partition of V.

Measures for Dispersion

With respect to the desire for well dispersed territories, unfortunately there is no clear definition of dispersion. As we did not find any literature on dispersion measures for territories, we decided to facilitate an approach from obnoxious location theory, see (Erkut and Neuman, 1991): We measure the dispersion of a territory by means of the pairwise distances of all basic areas assigned to that corporation for at least one of the two products. The goal is then to find a solution that maximizes the dispersion. Now, two questions arise. Firstly, as we have two-dimensional basic areas, how do we measure distances, and secondly, how to aggregate the pairwise distances. Concerning the first question, we decided to use the Euclidean distance between the polygonal representations of the basic areas to determine the pairwise distances. In that way, neighboring basic areas have distance zero. With respect to the second question, in the literature, usually either the sum of distances or the minimal distance is considered. To decide whether this approach yields suitable results and which of these two functions leads to better dispersed territories, we did some computational tests comparing the solutions visually as well as numerically. For a solution $X = \{X_1, \dots, X_m\}$, the first objective, called *median* objective, is made up of the sum over all pairwise distances: $f(X) = \sum_{k \in C} \sum_{i,j \in X_k} d_{ij}$. The second objective, denoted as *center* objective, determines over all corporations the smallest pairwise distance within a territory: $g(X) = \min_k \min_{i,j \in X_k} d_{ij}$. The corresponding solutions are called *maxisum* and *maximin*, respectively.

A visual comparison between two solutions for a problem with 100 basic areas (German zip code areas), six corporations, no splitting allowed, and identical parameters is given in Figure 1. On the left-hand side (right-hand side) we depict the maxisum (maximin) solution. We observe that the maximin solution yields well dispersed territories which are also much better than the maxisum solution which tends to build clusters of basic areas assigned to the same corporation. For other problem instances, similar outcomes can be noticed.

[Figure 1 about here.]

We also compared the two objectives numerically. For that, we denote with X^*_{sum} and X^*_{min} the territories of the optimal solutions obtained by maximizing $f(\cdot)$ and $g(\cdot)$, respectively. We took five different problem settings: 30 and 40 basic areas, and 3, 4, and 5 territories (five instances each), and determined the corresponding optimal solutions with respect to the two objectives. First we compare the different solutions with respect to the median objective. For each instance, we computed the ratio $f(X^*_{min})/f(X^*_{sum})$ between the median objective value of the maximin and maxisum solutions. Then we calculated the minimal (Min), maximal (Max), and average (Avg) ratio over all instances with the same parameter setting. The results are given in Table 1 under the heading “Median objective”.

[Table 1 about here.]

We observe that, with respect to the median objective, the maximin solutions perform quite well compared to the corresponding maxisum solutions. On average, their objective values are between six and ten percent below the optimal ones for the median objective. The reverse, however, is not true. The right hand side of the table gives the results when we compare the different solutions with respect to the center objective. The ratio in this case is computed as $g(X^*_{sum})/g(X^*_{min})$. In terms of the center objective, maxisum solutions perform poorly compared to the maximin solutions. Therefore, we decided to use the smallest pairwise distance, namely the center objective, and maximize this value to obtain well dispersed territories. Note that a similar observation was made in (Erkut and Neuman, 1991).

Mathematical Model

Next, we will state a first mathematical model for the problem. For that we need four sets of decision variables. For $i, j \in V$, $k \in C$, and $p = 1, 2$, let

$$x^p_{ik} = \begin{cases} 1 & \text{if basic area } i \text{ is assigned to corporation } k \text{ for product } p, \\ 0 & \text{otherwise.} \end{cases}$$

$$y_{ik} = \begin{cases} 1 & \text{if area } i \text{ is assigned to corp. } k \text{ for at least one of the two products,} \\ 0 & \text{otherwise.} \end{cases}$$

$$z_{ijk} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are both assigned to } k \text{ for at least one of the two products,} \\ 0 & \text{otherwise.} \end{cases}$$

$$e_i = \begin{cases} 1 & \text{if area } i \text{ is assigned to different corporations for products 1 and 2,} \\ 0 & \text{otherwise.} \end{cases}$$

As the z_{ijk}-variables are symmetric with respect to i and j, i.e., $z_{ijk} = z_{jik}$, we can assume w.l.o.g. that $i < j$. The mathematical model is now as follows:

$$\max \quad u$$

$$s.t. \quad \sum_{k \in C} x^p_{ik} = 1 \qquad \forall i \in V,\ p = 1, 2 \tag{1}$$

$$\sum_{i \in V} w_i\, x^p_{ik} \geq W\,(1-\tau)\, MS^p_k \qquad \forall k \in C,\ p = 1, 2 \tag{2}$$

$$\sum_{i \in V} w_i\, x^p_{ik} \leq W\,(1+\tau)\, MS^p_k \qquad \forall k \in C,\ p = 1, 2 \tag{3}$$

$$\sum_{i \in V_q} x^p_{ik} \geq |V_q|\,(1-\beta)\, MS^p_k \qquad \forall k \in C,\ p = 1, 2,\ q \in Q \tag{4}$$

$$\sum_{i \in V_q} x^p_{ik} \leq |V_q|\,(1+\beta)\, MS^p_k \qquad \forall k \in C,\ p = 1, 2,\ q \in Q \tag{5}$$

$$x_{ik}^1 - x_{ik}^2 \leq e_i \qquad \forall i \in V,\, k \in C \tag{6}$$

$$x_{ik}^2 - x_{ik}^1 \leq e_i \qquad \forall i \in V,\, k \in C \tag{7}$$

$$\sum_{i \in V} e_i \leq S \tag{8}$$

$$u \leq d_{ij}\, z_{ijk} + D(1 - z_{ijk}) \qquad \forall i, j \in V, i < j,\, k \in C \tag{9}$$

$$y_{ik} \leq x_{ik}^1 + x_{ik}^2 \qquad \forall i \in V,\, k \in C \tag{10}$$

$$x_{ik}^p \leq y_{ik} \qquad \forall i \in V,\, k \in C,\, p = 1, 2 \tag{11}$$

$$y_{ik} + y_{jk} \leq 1 + z_{ijk} \qquad \forall i, j \in V, i < j,\, k \in C \tag{12}$$

$$z_{ijk} \leq y_{ik} \qquad \forall i, j \in V, i < j,\, k \in C \tag{13}$$

$$z_{ijk} \leq y_{jk} \qquad \forall i, j \in V, i < j,\, k \in C \tag{14}$$

$$x_{ik}^p,\, y_{ik},\, z_{ijk} \in \{0, 1\} \qquad \forall i, j \in V,\, k \in C$$

Constraints (1) enforce that each basic area is assigned to a corporation for each of the two products. To ensure that the number of households is proportional to the market share for both products we use Constraints (2) and (3), where $\tau \in [0, 1]$ has to be chosen suitably. In an analogous fashion, the following two sets of constraints enforce a fair distribution of good, mediocre, and bad basic areas to corporations. Although allowing split basic areas, we bound their number from above by a predetermined value S using Constraints (6), (7), and (8). Note that we could alternatively add this criterion to the objective function, resulting in a bi-objective formulation. Finally, Constraints (9) through (14) together with the objective function model the center objective, where $D = \max_{i,j} d_{ij}$. (Formally we have $z_{ijk} = y_{ik} \cdot y_{jk}$ and $y_{ik} = 1$ iff $x_{ik}^1 + x_{ik}^2 \geq 1$.) Note that, in principle, we can drop the y-variables in the modeling of the objective function.

We call this problem the *Maximum Dispersion Territory Design Problem (MaxD-TDP)*. For the currently foreseen problem size (450 basic areas and 6-7 corporations) the model contains over 600 000 binary variables as well as constraints. Note that the MaxD-TDP is NP-hard, since we can reduce the PARTITION PROBLEM to it (set $C = \{1, 2\}$, $S = 0$, $MS_k^p = 0.5$, $\tau = 0$, $\beta = 1$, and $d_{ij} = 1$).

Remark. With respect to the aforementioned “value” of the waste returned, we would have to include constraints that are analogous to the constraints to enforce a fair distribution of good, mediocre, and bad basic areas to corporations.

Solving the Maximum Dispersion Territory Design Problem with GRASP

In this section we propose a GRASP heuristic for solving the MaxD-TDP. GRASP (Greedy Randomized Adaptive Search Procedure) is a multi-start metaheuristic for combinatorial problems, in which each iteration consists basically of two phases: construction and local search. The construction phase builds a (feasible) solution, whose neighborhood is investigated until a local minimum is found during the local search phase. The best overall solution is kept as the result. It is a well-known metaheuristic that captures good features of both pure greedy algorithms and random construction procedures, see (Feo and Resende, 1995). It has been successfully used for solving many combinatorial optimization problems and, in particular, applied to some commercial territory design problems, see (Ríos-Mercado and Fernández, 2009; Caballero-Hernández et al., 2007; Ríos-Mercado, 2007).

[Figure 2 about here.]

Algorithm 2 illustrates a generic GRASP implementation in pseudocode. The algorithm takes as an input an instance of the MDTDP, the maximum number of GRASP iterations, and the *quality parameter* α for building the restricted candidate list (RCL). In the following, we first discuss different construction heuristics before turning to the local search phase.

Construction Phase

For the construction phase of the GRASP heuristic we consider three different schemes to build initial solutions. Two of them (Procedures 1 and 1R) try to assign basic areas that are relatively close to each other to different corporations. The third one (Procedure 2) is based on the complementary idea of trying to assign to the same corporation basic areas that are relatively far away from each other. These procedures are now described in detail.

Construction Procedure 1

After sorting the pairwise distances in non-decreasing order, we go through this list step by step, starting with the smallest non-zero value. At a given iteration, the next largest distance d_{ij} is considered and if i and/or j are yet unassigned, we try to allocate them to different corporations. The allocation decision is hereby based on a greedy function that takes a distance-based measure and the sum of the relative violations of the upper balancing constraints into account. More precisely, if $X = (X_1, \ldots, X_m)$ is the partial solution obtained so far, for a given basic area i, a corporation k, and a product p, the relative violation of the upper bound of the market share balancing constraints (3) and the upper bound of the logistics quality balancing constraints (5) is given as

$$G_i^p(k) = \max\left\{\frac{w(X_k) + w_i}{W \cdot MS_k^p} - (1+\tau),\, 0\right\} + \max\left\{\frac{c^{q_i}(X_k) + 1}{|V_{q_i}| \cdot MS_k^p} - (1+\beta),\, 0\right\}$$

where q_i is the logistics index of basic area i, $w(X_k) := \sum_{j \in X_k} w_j$, and $c^{q_i}(X_k) := |X_k \cap V_{q_i}|$. Note that it makes no sense to include the lower bounds, as they will always be violated until the very end of the construction procedure. The greedy function is then defined as $\phi_i(k) = \lambda F_i(k) - (1-\lambda) G_i(k)$, where $\lambda \in [0,1]$, $G_i(k) = G_i^1(k) + G_i^2(k)$ and $F_i(k) = d_i(X_k) := \min_{j \in X_k} d_{ij}$ (in case $X_k = \emptyset$, then $d_i(X_k) := \infty$).

[Figure 3 about here.]

Figure 3 shows the pseudocode of Construction Procedure 1. DL denotes the list of all pairwise distances ordered by non-decreasing values, with ties arbitrarily broken, and $D^{(r)}$ denotes the r-th element of the list. The restricted candidate list is defined as RCL $= \{k : \phi_i(k) \geq \phi_i^{\max} - \alpha(\phi_i^{\max} - \phi_i^{\min})\}$, where $\phi_i^{\min} = \min_k \phi_i(k)$ and $\phi_i^{\max} = \max_k \phi_i(k)$.

Construction Procedure 1R

As the upper bound constraints (3) and (5) have been taken into account only implicitly through the greedy function in the previous procedure, we likely obtain infeasible solutions. Therefore, in a variant of Construction Procedure 1, we will try to obtain solutions with no or at least fewer upper bound violations. Construction Procedure 1R has two phases. In the first one, basic areas are assigned to corporations as in Construction Procedure 1, provided that they do not violate (3) and (5). This can be achieved easily through a modified restricted candidate list: $RCL = \{k : G_i(k) = 0 \wedge \phi_i(k) \geq \phi_i^{\max} - \alpha(\phi_i^{\max} - \phi_i^{\min})\}$. As some basic areas may remain unassigned at the termination of the first phase, these areas are assigned to corporations in a second phase in a different fashion. Throughout this phase, to reduce the violation of (3) and (5), we use as greedy function just $G_i(k)$. Splitting of basic areas is allowed in the second phase when no more than $|S|$ basic areas remain unassigned. The procedure is shown is Figure 4.

[Figure 4 about here.]

Construction Procedure 2

Using a different rationale, we now try to assign to the same corporation basic areas that are relatively distant from each other. The procedure consists of two phases and is shown in Figure 5. In the first phase the iterative procedure builds m territories, one at a time, using a farthest insertion greedy function that assigns to the territory X_k currently being built basic areas relatively "far away" from X_k (lines 7 to 11). Since, again, no violation of the upper bound balancing constraints is allowed, the RCL only contains unassigned basic areas for which $G_i(k) = 0$, see line 8 (note that $\phi^{max} = \max_i \phi(i)$ and $\phi^{min} = \min_i \phi(i)$). If some basic areas remain unassigned at the termination of this phase, we proceed to a second phase that is exactly as in Construction Procedure 1R.

[Figure 5 about here.]

Local Search

After the construction phase, which may yield an infeasible solution, a local search phase is applied. In this phase we attempt to recover feasibility as well as to improve the objective function value. Solutions are now evaluated by means of a function that weighs both infeasibility with respect to the balancing constraints as well as the smallest pairwise distance among the territories. This function is similar to the greedy function $\phi_i(k)$ used in the construction phase, however, with the addition that now the sum of relative infeasibilities $G_i(k)$ takes into account not only the upper bound balancing constraints, but also the violation of the lower bound balancing constraints (2) and (4). The types of exchange moves that we consider for the local search are the following:

Type A1: For all products, reassign a basic area i that is currently assigned to some territory X_k of corporation k to the territory $X_{k'}$ of a different corporation $k' \neq k$. The size of the neighborhood is $n \times m$.

Type A2: Reassign a basic area i just for product p from its current territory to a different corporation's territory. Splitting is allowed. The size of the neighborhood is $2n \times m$.

Type B: Exchange the assignment of basic areas i and j currently allocated to different corporations for one or both products. The size of the neighborhood is $2n^2$.

Computational Results

For the computational tests we generated problem instances using real-world data obtained from the GIS *ArcView* (`www.esri.com`, 2009). Basic areas correspond to German zip-code areas with their respective number of households. The instances range from 100 up to 300 basic areas in steps of 50, and with four up to seven corporations. To generate an instance, we extracted all zip-code areas lying within a rectangle. The instances represent rural, urban as well as mixed regions. We generated five instances for each number of basic areas, except for the last, where we have just four instances. In combination with the four different numbers of corporations, this yields in total 96 instances. The tolerances for the demands and for the logistics indices are $\tau = 0.05$ and $\beta = 0.2$, respectively. The pairwise distances were computed as the Euclidean distance between the polygonal representations of the zip-code areas. In that way, neighboring basic areas have distance zero. The logistics indices were chosen randomly such that we have approximately the same number of good, mediocre, and bad basic areas. Unfortunately, a suitable value S is not explicitly stated anywhere. Therefore, we set the maximal number S of split areas to 20% of the number n of basic areas. Finally, the market shares of the companies are computed independently for the two products. A market share is drawn uniformly from the interval $[\frac{0.75}{m}, \frac{1.25}{m}]$. At the end, the market shares are normalized to obtain a total sum of 1.

The heuristic procedures were coded in C++ and compiled with the Sun C++ compiler workshop 8.0 under the Solaris 9 operating system. They were run on a SunFire V440 with 8 GB of RAM and 4 UltraSparc III procesors at 1062 MHz.

Parameter Tuning

In the first part of the computational experiments we focus on tuning the parameters of the GRASP heuristic, namely the RCL parameter α and the weight parameter λ of the greedy function ϕ. To this end, we run the different versions of the GRASP with an iteration limit of 500 and no local search phase, for measuring both the degree of violation of the balancing constraints and the value of the objective function. The set of values for λ is $\{0.5, 0.6, \ldots, 1.0\}$, and for α we consider the set $\{0, 0.2, \ldots, 1.0\}$. Figures 6, 7, 8, and 9, display results for Construction Procedure 1 with λ fixed, Procedure 1 with α fixed, Procedure 1R, and Procedure 2, respectively. In these figures, the left vertical axis measures the average (over all instances) relative infeasibility with respect to the upper and lower balancing constraints, whereas the right vertical axis measures the average (over all instances) percent gap between the objective function value of the obtained solution and the best solution found with all the values of the tested parameters.

For Construction Procedure 1, a value of $\alpha = 0.2$ consistently finds the best compromise between infeasibility and deviation from maximum dispersion for each tested value of λ. Comparing the values of λ we can observe that $\lambda = 0.5$ yields a good compromise. Due to their mechanism for building solutions, the λ parameter plays a minor role in Procedures 1R and 2. Thus, only the parameter α was evaluated. We can observe that $\alpha = 0.2$ and $\alpha = 0.8$ yielded the best compromise between deviation from maximum dispersion and infeasibility for each of these procedures. Comparing Figures 6 and 8 shows that the relative infeasibilities for Procedure 1R are lower than for Procedure 1 with no distinguishable differences in the percentage gaps, justifying the variation.

[Figure 6 about here.]

[Figure 7 about here.]

[Figure 8 about here.]

[Figure 9 about here.]

It is worth noting that in all the above tests no feasible solutions (with respect to the balancing constraints) were found. This reinforces the role of the local search phase in the overall solution method.

Local Search Strategies

Next, we investigate the behavior and effect of the local search. In the previous section we introduced three different neighborhoods. For our tests we consider three different combinations of these neighborhoods: LS1 (apply type A1 and then type A2 neighborhood), LS2 (apply type A2 and then type A1 neighborhood), and finally LS3 (apply type A2, then type A1, and then type B neighborhood). Since the neighborhoods are polynomially bounded and not too large, we use a best improvement strategy. For Constructive Procedure 1, we use $\alpha = 0.2$ and $\lambda = 0.5$, for Procedure 1R, we set $\alpha = 0.2$ and $\lambda = 1.0$, and for Procedure 2, $\alpha = 0.8$ and $\lambda = 1.0$ was used. The iteration limit was set to 2000 GRASP iterations.

Results for Constructive Procedures 1, 1R, and 2 with LS1 and LS2 are displayed in Table 2. Note that in this table, for a particular construction procedure, the relative dispersion gap is computed with respect to the best known value of the dispersion objective found by any of the local search strategies for that particular heuristic. So, in this regard, only a comparison between local search strategies (and not among heuristics) is valid. A comparison among the heuristics is made in the following section.

[Table 2 about here.]

The results in Table 2 indicate that, independently of the construction procedure, both strategies LS1 and LS2 were able to obtain always feasible solutions, which represents a significative improvement with respect to the construction phase where no feasible solutions were found (although solutions with relatively small deviations from feasibility were obtained). This shows the effectiveness of the local search strategies for repairing the infeasibility issue. The results also indicate that strategy LS2 outperforms strategy LS1 in terms of relative deviations from the best known value for the dispersion objective. LS2 was consistently better than LS1 for every value of n tested. In addition, LS2 uses considerably less CPU time than LS1 (about 40% less in average). Therefore, we can conclude that it is more advantageous to explore A2 first, and then A1.

Heuristic Comparison

Table 3 shows a comparison among the heuristics under local search strategy LS2. This time the relative dispersion gap is computed with respect to best known value of the dispersion objective found by any procedure under any local search strategy. We observe that, in terms of solution quality, Construction Procedure 2 exhibits a very poor performance compared to the other two. Procedures 1 and 1R perform very similarly with respect to both dispersion objective and CPU time, with Procedure 1 being slightly better than 1R in terms of the dispersion objective. In particular, for the largest problems (300 basic areas), both heuristics find the same solution for 75% of the instances. For problem instances below 300 basic areas, Construction Procedure 1 finds the best solution. Overall, Procedures 1 and 1R each find 76 best solutions. It is important to highlight the tremendous benefit reported by the local search phase in each case. Most of the solutions found in the construction phase were infeasible (feasibility success of 5% and large deviations from feasibility in some cases). Neighborhoods A1 and A2 helped to bring this figure up to 100.0%. This suggests further work on the local search schemes could be worthwhile.

[Table 3 about here.]

We also tested strategy LS3 for the 300-node instances with Procedure 1. For 11 out of 16 instances, no improvement was found. Moreover, while the overall average relative improvement is less than 1.5%, the average CPU time for LS3 is 19778.9 seconds, which represents, compared to the 1056.5 seconds for LS2, a very large increase that barely pays off in terms of solution quality.

We now perform a comparison between the heuristic and the exact solutions for a set of small instances. We generated five instances each for the parameter combinations $(n, m) = (20, 2\&3)$ and $(40, 3\&4)$. To obtain optimal solutions, we implemented the mathematical model using ILOG Concert and solved it with CPLEX 11.0 with a time limit of one hour on an Intel Core2 with 1.86 GHz and 2GB Ram (www.ilog.com, 2009). For the heuristic solutions we used a LS3 local search strategy for the three of them, and 2000 GRASP iterations. For Procedure 1, 1R, and 2, we set α and λ to $(0.2, 0.5)$, $(0.2, 1.0)$, and $(0.8, 1.0)$, respectively.

[Table 4 about here.]

In Table 4 we present the results of this comparison for each instance. For the exact method we are displaying the best value of objective function found (Obj), the relative optimality gap (OpGap) in %, and the CPU time (T) in seconds. It is clear that an optimality gap of 0 means an optimal solution was found. For each of the heuristics we give the relative gap (ReGap) in % between the solution found by the heuristics and the solution found by the exact method. A value of “inf” in this column means that the heuristic failed to find a feasible solution. First note that for the instances with 20 basic areas, the exact algorithm was able to solve optimally all of them in a reasonable amount of time (less than a minute). Nevertheless, Heuristic 2 found 9 of 10 optimal solutions in this data set. Now, for the larger instances, the exact method begins to struggle. Only three optimal solutions were found, and for the other 7, after 1 hr of running time, optimality gaps were extremely large. Note that the heuristics found solutions equal or very close to those found by the exact method. In fact, heuristic 2 matched the best exact method solution in 7 out of 10 cases. This was done in an effort of less than a minute. Note that although the maximal relative gap is 20%, on average this gap is 2.8% and therefore acceptable.

GRASP Convergence

Another issue regarding GRASP-based heuristics is to determine a suitable value for the number of iterations. Figure 10 displays the number of best solutions found within a given range of GRASP iterations. Four ranges are plotted: $[1, 500]$, $[501, 1000]$, $[1001, 1500]$, and $[1501, 2000]$, in all cases with LS2. For example, the bars on the 200-node instances mean that using Construction Procedure 1 for these 20 instances, the iteration in which the best solution was found was 15 times in the range $[1, 500]$, 1 time in the range $[501, 1000]$, 2 times in the range $[1001, 1500]$, and 2 times in the range $[1501, 2000]$. As can be seen, for both construction procedures around 74% of the best solutions were found in the first 500 iterations, and 81% were found in less than 1000 iterations. Still only 10% of the best solutions required above 1500 iterations. Table 5 summarizes the totals for the two heuristics. Values in each row display the total frequency over all instances for each heuristic.

[Figure 10 about here.]

[Table 5 about here.]

Conclusions

In this paper we introduced the Maximum Dispersion Territory Design Problem, motivated by the adaptation of the German system to the 2003 recycling directive WEEE of the EC for one type of products. The problem consists of designing territories for the involved corporations that re-collect and recycle different types of white goods at the end of their lifetime. To avoid monopolistic concentration, the collection stations assigned to a corporation should be evenly dispersed over Germany. Moreover, the assignment of stations to corporations should be made such that the average amount of returned items is proportional to the market share of the company. To the best of our knowledge, this is the first model based on maximum dispersion in the territory design literature.

We have presented the problem and given a mathematical programming formulation which, unfortunately, can only solve small instances using standard solvers. Therefore, we have proposed a solution approach based on the GRASP methodology. Different criteria, both for the construction and for the local search processing phase of the GRASP heuristic, have been analyzed. Extensive computational experiments have been done in order to identify the best strategies and the most appropriate parameter values. The obtained results are satisfactory, since we are able to obtain good quality results in small computation times.

There are several challenging avenues for future research. Among others, these include the design of more sophisticated metaheuristics, as well as the study of stronger mathematical programming formulations. Moreover, at a longer term, we want to combine the design of the territories with the planning of the collection routes.

Acknowledgments This research has been partially supported through grants MTM2006-14961-C05-01, of the Spanish Plan Nacional de Investigación Científica, Desarrollo e Innovación Tecnológica (I+D+I), SEP-CONACYT 48499-Y, of the Mexican National Council for

Science and Technology, and PAICYT CA-1478-07, of the Universidad Autónoma de Nuevo León through its Scientific and Technological Research Support Program. These supports are gratefully acknowledged.

References

Caballero-Hernández, S. I., R. Z. Ríos-Mercado, F. López, and S. E. Schaeffer (2007, November). Empirical evaluation of a metaheuristic for commercial territory design with joint assignment constraints. In J. E. Fernandez, S. Noriega, A. Mital, S. E. Butt, and T. K. Fredericks (Eds.), *Proceedings of the 12th Annual International Conference on Industrial Engineering Theory, Applications, and Practice (IJIE)*, Cancun, Mexico, pp. 422–427. ISBN: 978-0-9654506-3-8.

Erkut, E. and S. Neuman (1991). Comparison of four models for dispersing facilities. *INFOR 29*(2), 68–86.

Feo, T. A. and M. G. C. Resende (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization 6*(2), 109–133.

Grunow, M. and C. Gobbi (2008). A MILP for configuring reverse networks for electric and electronic waste. Presentation at the OR 2008 conference, Augsburg, Germany.

Hischier, R., P. Wäger, and J. Gauglhofer (2005). Does WEEE recycling make sense from an environmental perspective?: The environmental impacts of the swiss take-back and recycling systems for waste electrical and electronic equipment (WEEE). *Environmental Impact Assessment Review 25*(5), 525–539.

Kalcsics, J. (2006). *Unified approaches to territory design and facility location.* Series in Operations Research. Shaker-Verlag, Aachen. PhD thesis.

Kalcsics, J., S. Nickel, and M. Schröder (2005). Towards a unified territorial design approach - Applications, algorithms and GIS integration. *TOP 13*(1), 1–56.

Queiruga, D., G. Walther, J. González-Benito, and T. Spengler (2008). Evaluation of sites for the location of WEEE recycling plants in Spain. *Waste Management 28*(1), 181–190.

Ríos-Mercado, R. Z. (2007, October). Computational experience with a reactive GRASP for a large scale commercial territory design problem. In M. J. Geiger and W. Habenicht (Eds.), *Proceedings of EU/ME 2007 Metaheuristics in the Service Industry*, Stuttgart, Germany, pp. 72–79. ISBN: 978-3-00-022976-3.

Ríos-Mercado, R. Z. and E. A. Fernández (2009). A reactive GRASP for a commercial territory design problem with multiple balancing requirements. *Computers & Operations Research 36*(3), 755–776.

Turner, M. and D. Callaghan (2007). UK to finally implement the WEEE directive. *Computer Law & Security Report 23*(1), 73–76.

WEEE (2003). Directive 2002/96/EC of the European Parliament and of the Council of 27 January 2003 on waste electrical and electronic equipment (WEEE) - Joint declaration of the European parliament, the Council and the Commission relating to Article 9. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32002L0096:EN:HTML.

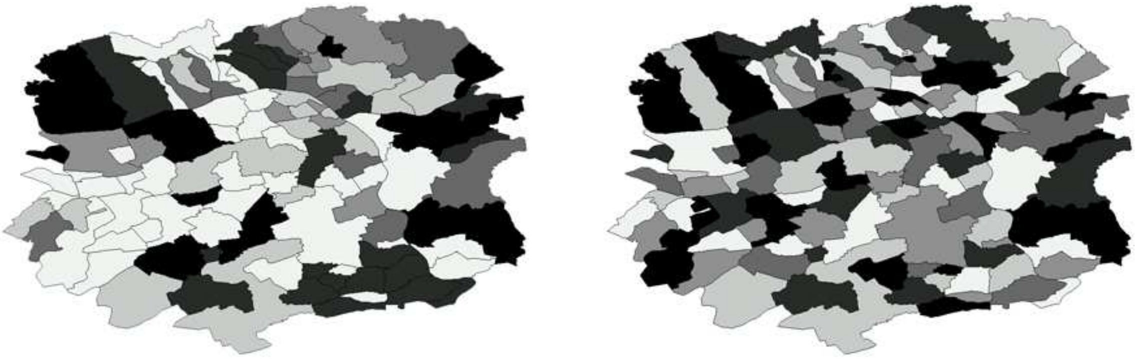

Figure 1: Solutions obtained by maximizing the sum of distances (left-hand side) and by maximizing the minimal distance (right-hand side).

```
function GRASP ($limit\_iterations$, $\alpha$)
Input: $limit\_iterations$ := GRASP iteration limit;
       $\alpha$ := GRASP RCL quality parameter,
Output: A feasible assigment $X^{\mathrm{best}}$.

$X^{\mathrm{best}} \leftarrow \emptyset$;
for ($l = 1, \ldots,$ $limit\_iterations$) do
    $X \leftarrow$ ConstructGreedyRandomized($\alpha$);
    $X \leftarrow$ LocalSearch($X$);
    if ($X$ better than $X^{\mathrm{best}}$) then $X^{\mathrm{best}} \leftarrow X$;
endfor;
return $X^{\mathrm{best}}$;
end GRASP
```

Figure 2: GRASP pseudocode for MDTDP.

```
function ConstructGreedyRandomized_1 ($\alpha$)
Input: $\alpha :=$ GRASP RCL quality parameter.
Output: An assignment $X = \{X_k\}_{k \in C}$.

0   $X_k = \emptyset, k \in C$; $V^a = \emptyset$ (set of assigned basic areas);
1   sort $\{d_{ij}\}$, $i \neq j$, and store them in $DL$;
2   while ($|V^a| < |V|$ AND $|DL| > 0$) do
3       $d_{ij} \leftarrow D^{(1)}$ (first element in $DL$); $DL \leftarrow DL \setminus \{D^{(1)}\}$;
4       if ($i \in V^a$ AND $j \in V^a$) go to step 2;
5       if ($i \notin V^a$ AND $j \in V^a$) [Note: opposite case is symmetric]
6           compute greedy function $\phi_i(k)$, $k \in C$;
7           build RCL; $\hat{k} \leftarrow$ rand(RCL);
8           if ($\hat{k} =$ territory of $j$) then RCL $\leftarrow$ RCL$\setminus\{\hat{k}\}$; $\hat{k} \leftarrow$ rand(RCL); endif;
9           assign $i$ to corporation $\hat{k}$: $X_{\hat{k}} \leftarrow X_{\hat{k}} \cup \{i\}$; $V^a \leftarrow V^a \cup \{i\}$;
10      endif;
11      if ($i \notin V^a$ AND $j \notin V^a$)
12          compute greedy function $\phi_i(k)$; $k \in C$
13          build RCL; $\hat{k} \leftarrow$ rand(RCL);
14          assign $i$ to corporation $\hat{k}$: $X_{\hat{k}} \leftarrow X_{\hat{k}} \cup \{i\}$; $V^a \leftarrow V^a \cup \{i\}$;
15          compute greedy function $\phi_j(k)$, $k \in C$;
16          build RCL; $k' \leftarrow$ rand(RCL);
17          if ($k' = \hat{k}$) then RCL $\leftarrow$ RCL $\setminus\{k'\}$; $k' \leftarrow$ rand(RCL); endif;
18          assign $j$ to company $k'$: $X_{k'} \leftarrow X_{k'} \cup \{j\}$; $V^a \leftarrow V^a \cup \{j\}$;
19      endif;
20  endwhile;
21  return $X = \{X_1, \ldots, X_m\}$;
end ConstructGreedyRandomized_1
```

Figure 3: Construction Procedure 1.

```
function ConstructGreedyRandomized_1R ($\alpha$)
Input: $\alpha$ := GRASP RCL quality parameter.
Output: A "feasible" assignment $X$.

0      $X_k = \emptyset, k \in C$; $\widehat{X}_k^p = \emptyset, k \in C, p = 1, 2$; $V^a = \emptyset$ (set of assigned basic areas);
1      sort $\{d_{ij}\}$ and store them in $DL$;
2-20   as in Construction Procedure 1 but with a modified RCL (see text);
21     while ($|V \setminus V^a| > S$) do
22         select $j'$ and $k'$ such that $G_{j'}(k') = \min\{G_i(k) \ : \ k \in C, i \in V \setminus V^a\}$;
23         $X_{k'} \leftarrow X_{k'} \cup \{j'\}$; $V^a \leftarrow V^a \cup \{j'\}$;
24     endwhile;
25     $X_k^p \leftarrow X_k, \forall p \in P, k \in C$;
26     while ($|V^a| < |V|$) do
27         select $j'$, $k'_1$, $k'_2$ such that
           $G^1_{j'}(k'_1) + G^2_{j'}(k'_2) = \min\{G^1_i(k_1) + G^2_i(k_2) \ : \ k_1, k_2 \in C, i \in V \setminus V^a\}$;
28         $X^p_{k'_p} \leftarrow X^p_{k'_p} \cup \{j' \ \}$ and $X_{k_p} \leftarrow X_{k_p} \cup \{j'\}, \ p \in P$;
29         $V^a \leftarrow V^a \cup \{j'\}$;
30     endwhile;
31     return $X = \{X_k^p\}, k \in C, p \in P$;
end ConstructGreedyRandomized_1R
```

Figure 4: Construction Procedure 1R.

```
function ConstructGreedyRandomized_2 ($\alpha$)
Input: $\alpha$ := GRASP RCL quality parameter.
Output: A “feasible” assignment $X$.

0  $X_k = X_k^p = \emptyset, k \in C, p = 1,2$; $V^a = \emptyset$ (assigned basic areas); $k \leftarrow 1$;
1  while ($k \leq |C|$) do
2        if ($k = 1$) then $X_k = \{i\}$ for $i \in V$ arbitrarily chosen;
3        else choose $i' = \arg\min_{i \in V \setminus V^a} d(i, X_1 \cup \ldots \cup X_{k-1})$; $X_k \leftarrow X_k \cup \{i'\}$;
4        endif;
5        $V^a \leftarrow V^a \cup \{i'\}$;
6        while ( $G_i(k) = 0$ for some $i$ ) do
7              compute greedy function $\phi(i) = d(i, X_k)$, $i \in V \setminus V^a$;
8              build RCL $= \{i \in V \setminus V^a \ : \ G_i(k) = 0 \wedge \phi(i) \geq \phi^{\max} - \alpha(\phi^{\max} - \phi^{\min})\}$;
9              if ( RCL $= \emptyset$ ) go to step 15;
10             $j \leftarrow$ rand(RCL);
11             assign $j$ to company $k$: $X_k \leftarrow X_k \cup \{j\}$; $V^a \leftarrow V^a \cup \{j\}$;
12       endwhile;
13       $k \leftarrow k + 1$;
14 endwhile;
15 apply second phase as in lines 21-30 of Construction Procedure 1R
16 return $X = \{X_k^p\}, k \in C, p \in P$;
end ConstructGreedyRandomized_2
```

Figure 5: Construction Procedure 2.

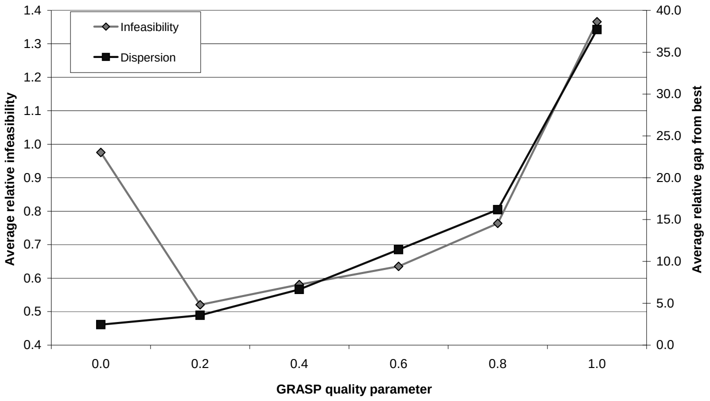

Figure 6: Construction Procedure 1 with $\lambda = 0.5$ as a function of GRASP quality parameter (α).

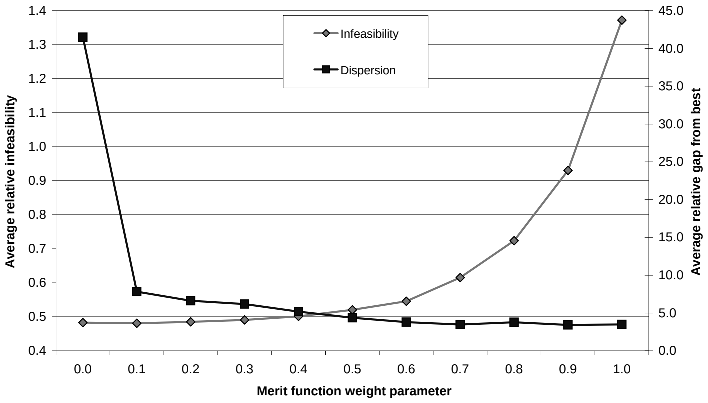

Figure 7: Construction Procedure 1 with $\alpha = 0.2$ as a function of the merit function weight parameter (λ).

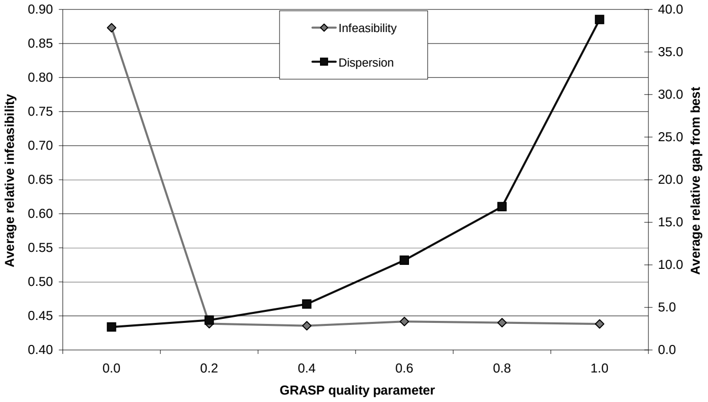

Figure 8: Construction Procedure 1R with $\lambda = 1.0$ as a function of the GRASP quality parameter (α).

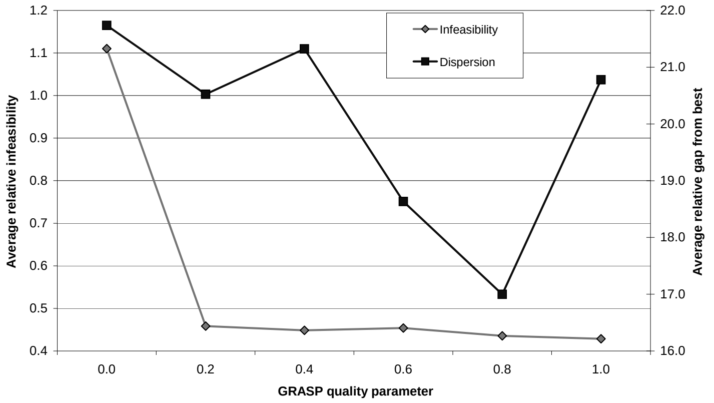

Figure 9: Construction Procedure 2 with $\lambda = 1.0$ as a function of the GRASP quality parameter (α).

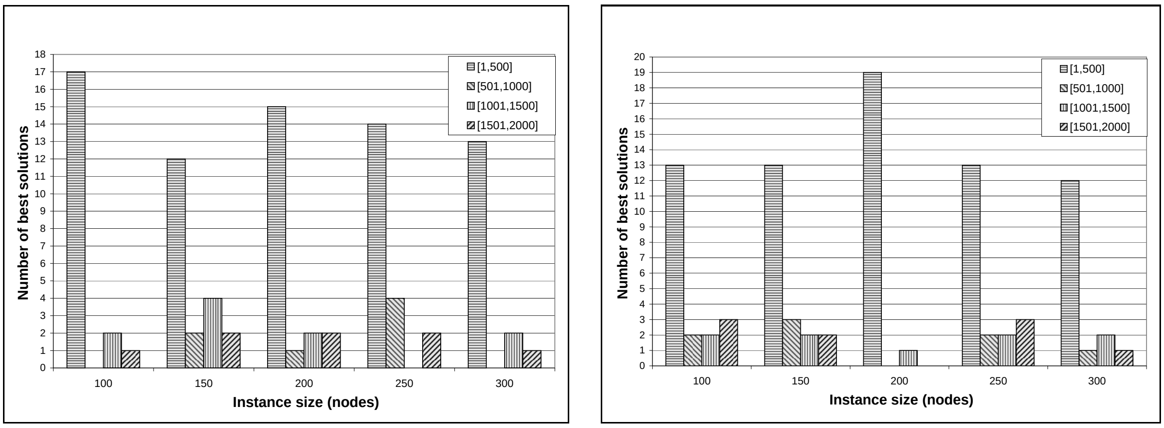

Figure 10: Convergence of Procedure 1 (left-hand side) and Procedure 1R under LS2.

		Median objective ratio			Center objective ratio		
		$f(X^*_{min})/f(X^*_{sum})$			$g(X^*_{sum})/g(X^*_{min})$		
n	m	Avg	Min	Max	Avg	Min	Max
30	3	0.91	0.88	0.94	0.40	0.06	0.54
30	4	0.93	0.89	0.97	0.32	0.13	0.50
40	3	0.90	0.83	0.93	0.22	0.15	0.33
40	4	0.92	0.88	0.94	0.25	0.10	0.50
40	5	0.94	0.85	0.99	0.29	0.14	0.42

Table 1: Numerical comparison of the maxisum and maximin solutions.

		Procedure 1		Procedure 1R		Procedure 2	
n	Statistic	LS1	LS2	LS1	LS2	LS1	LS2
100	No. of infeasible solutions	0	0	0	0	0	0
	Average relative dispersion gap (%)	2.7	0.0	1.5	0.5	2.8	0.8
	Worst relative dispersion gap (%)	15.9	0.0	11.8	5.2	15.3	6.3
	Average CPU time (in seconds)	173	128	157	120	185	173
150	No. of infeasible solutions	0	0	0	0	0	0
	Average relative dispersion gap (%)	3.5	1.4	2.3	0.0	4.3	0.9
	Worst relative dispersion gap (%)	14.8	21.1	11.2	0.0	12.6	6.0
	Average CPU time (in seconds)	396	265	361	257	461	440
200	No. of infeasible solutions	0	0	0	0	0	0
	Average relative dispersion gap (%)	2.4	0.0	4.6	0.2	5.3	0.7
	Worst relative dispersion gap (%)	15.1	0.0	25.3	3.6	20.7	7.9
	Average CPU time (in seconds)	676	440	597	411	823	787
250	No. of infeasible solutions	0	0	0	0	0	0
	Average relative dispersion gap (%)	8.4	0.0	5.4	0.2	3.8	1.1
	Worst relative dispersion gap (%)	17.3	0.0	12.7	3.3	13.7	9.6
	Average CPU time (in seconds)	1178	699	979	671	1464	1277
300	No. of infeasible solutions	0	0	0	0	0	0
	Average relative dispersion gap (%)	2.6	0.0	3.2	0.0	2.8	1.6
	Worst relative dispersion gap (%)	18.8	0.0	26.6	0.0	12.3	19.0
	Average CPU time (in seconds)	1728	1056	1645	1067	2383	2064
Total	No. of infeasible solutions	0	0	0	0	0	0
	Average relative dispersion gap (%)	4.0	0.3	3.4	0.2	3.9	1.0
	Worst relative dispersion gap (%)	18.8	21.1	26.6	5.2	20.7	19.0

Table 2: Effect of local search strategies in Constructive Procedure 1, 1R, and 2.

n	Statistic	Proc 1	Proc 1R	Proc 2
100	Avg. rel. dispersion gap (% from best known)	0.6	1.2	21.7
	Worst relative dispersion gap (%)	6.4	16.6	34.4
	Number of infeasible solutions	0	0	0
	Number of best solutions	14	16	0
	Average CPU time (in seconds)	128	120	173
150	Avg. rel. dispersion gap (% from best known)	0.4	1.3	19.4
	Worst relative dispersion gap (%)	4.2	8.8	28.5
	Number of infeasible solutions	0	0	0
	Number of best solutions	17	15	0
	Average CPU time (in seconds)	265	257	440
200	Avg. rel. dispersion gap (% from best known)	1.5	0.2	24.1
	Worst relative dispersion gap (%)	13.3	3.6	44.6
	Number of infeasible solutions	0	0	0
	Number of best solutions	15	18	1
	Average CPU time (in seconds)	440	411	787
250	Avg. rel. dispersion gap (% from best known)	0.7	0.7	28.4
	Worst relative dispersion gap (%)	3.2	6.2	45.6
	Number of infeasible solutions	0	0	0
	Number of best solutions	14	15	0
	Average CPU time (in seconds)	699	671	1277
300	Avg. rel. dispersion gap (% from best known)	0.0	0.9	30.7
	Worst relative dispersion gap (%)	0.0	7.6	53.2
	Number of infeasible solutions	0	0	0
	Number of best solutions	16	12	0
	Average CPU time (in seconds)	1056	1067	2064
Total	Avg. rel. dispersion gap (% from best known)	0.7	0.9	24.6
	Worst relative dispersion gap (%)	13.3	16.6	53.2
	Number of infeasible solutions	0	0	0
	Number of best solutions	76	76	0

Table 3: Comparison of construction procedures under LS2.

		Exact method			Heuristic 1 + LS3			Heuristic 1R + LS3			Heuristic 2 + LS3		
n	m	Obj	OpGap	T	Obj	ReGap	T	Obj	ReGap	T	Obj	ReGap	T
20	2	0.9	0	2	0.7	22.2	7	0.7	22.2	6	0.9	0.0	7
20	2	0.8	0	1	0.7	12.5	9	0.7	12.5	5	0.8	0.0	10
20	2	0.9	0	2	0.9	0.0	10	0.9	0.0	10	0.9	0.0	7
20	2	1.8	0	1	0.9	50.0	6	1.3	27.8	10	1.8	0.0	8
20	2	0.7	0	2	0.2	71.4	8	0.7	0.0	7	0.7	0.0	10
20	3	1.7	0	27	1.7	0.0	9	1.7	0.0	8	1.7	0.0	10
20	3	2.0	0	10	2.0	0.0	11	1.8	10.0	10	2.0	0.0	11
20	3	1.9	0	30	0.9	52.6	10	1.2	36.8	9	1.9	0.0	11
20	3	2.0	0	31	inf	50.0	8	1.3	35.0	9	1.6	20.0	10
20	3	1.5	0	25	1.5	0.0	10	1.1	26.7	8	1.5	0.0	11
40	3	1.0	0	321	0.8	20.0	58	0.8	20.0	46	1.0	0.0	68
40	3	1.5	0	259	1.4	6.7	71	1.4	6.7	54	1.4	6.7	58
40	3	1.3	85	3600	1.1	15.4	52	0.9	30.8	40	1.3	0.0	55
40	3	1.2	0	500	1.2	0.0	75	1.2	0.0	62	1.2	0.0	65
40	3	1.2	483	3600	1.0	16.7	63	1.0	16.7	51	1.2	0.0	50
40	4	0.8	412	3600	0.8	0.0	54	0.8	0.0	46	0.8	0.0	70
40	4	0.4	1000	3600	0.4	0.0	53	0.4	0.0	48	0.4	0.0	59
40	4	1.7	376	3600	1.6	5.9	62	1.7	0.0	55	1.4	17.6	62
40	4	1.7	210	3600	1.6	5.9	70	1.5	11.8	42	1.5	11.8	51
40	4	1.2	408	3600	1.2	0.0	64	1.2	0.0	37	1.2	0.0	62

Table 4: Heuristic comparison on small instances; T is given in seconds and OpGap and ReGap in %.

	Iteration range			
Procedure	$[0, 500]$	$[501, 1000]$	$[1001, 1500]$	$[1501, 2000]$
Construction Procedure 1	71	7	10	8
Construction Procedure 1R	70	8	9	9

Table 5: Best solution frequency analysis of GRASP under LS2 with respect to iteration limit.