Tabu Search with Strategic Oscillation for Improving Recollection Assignment Plans of Waste Electric and Electronic Equipment

Roger Z. Ríos-Mercado
Universidad Autónoma de Nuevo León
Graduate Program in Systems Engineering
E-mail: *roger.rios@uanl.edu.mx*

José L. González-Velarde
Tecnológico de Monterrey
Center for Quality and Manufacturing
E-mail: *gonzalez.velarde@itesm.mx*

Jabneel R. Maldonado-Flores
E-mail: *jabneelmf@gmail.com*

Technical Report PISIS–2017–01
Graduate Program in Systems Engineering
Department of Mechanical and Electrical Engineering
Universidad Autónoma de Nuevo León
San Nicolás de los Garza, NL, Mexico
30 May 2017

Abstract

This paper studies a districting problem arising in the recollection of waste of electrical and electronic equipment (WEEE). Given a set of recollection bins, where users return end-of-life electronic goods, located across a region or country, the design problem consists of assigning these bins to the companies who will be responsible for the recollection at a later stage. This assignment must meet certain planning and legal requirements such as a fair household distribution according to the market share of each company and a fair assignment based on the bin's infrastructure quality. According the the current WEEE II Directive, this assignment must be done in a such a way so as to avoid, to the best possible extent, regional monopolies. This is achieved by maximizing a dispersion function. A tabu search metaheuristic with advanced feature of strategic oscillation is proposed for this NP-hard combinatorial optimization problem. In addition, a few upper bounding schemes are developed and tested. The empirical work shows the effectiveness of the tabu search and each of its components over a wide set of instances from the literature.

Keywords: Combinatorial optimization; Districting; WEEE Recollection; Metaheuristics; Tabu search; Strategic oscillation.

1 Introduction

The production of electrical and electronic equipment is one of the markets of larger growth at world-wide level. This entails that the number of equipment of this type that falls in disuse will continue growing during the next decades. Rough estimates indicate that in the European Union the amount of waste of electrical and electronic equipment (WEEE) will increase from 3% to 5% per year.

Since 2003, given the WEEE directive, in the European Union recycling of electric home appliances is mandatory. Originally, the management of waste of electric and electronic equipment was regulated by the WEEE Directive (Directive 2002/96/EC). In August 2012, the European Union issued a revision of the WEEE Directive (Directive 2012/19/EU or, simply, the WEEE II Directive.) Within this scheme of recycling, a diversity of problems that can be tackled by means of operational research techniques has arisen; in particular this paper focuses in the design of the recollection territories that will be assigned to each company. The European law establishes that the original equipment manufacturers are responsible for WEEE recollection in a percentage that is proportional to the volume of its sales in the market. The authorities have established harvesting points where the users deposit the electric home appliances at the end of their useful life. The problem at hand consists of finding the best possible way to group the harvesting points. Each group, also called a territory, is in particular associated to a 3PL company that will be responsible for the recycling of the equipment there collected.

As well, each point of harvesting has associated a certain number of potential users and a quality level of infrastructure that can be good, mediocre or bad. In order to realize a right distribution of the collecting points, it is important that the defined territories approximately contain the same number of inhabitants and the same number of harvesting points of each quality level, that is to say, it is desirable to obtain territories balanced with respect to these criteria.

Another important characteristic of the problem is the need for classifying the disposed equipment into two different categories. The high toxicity of the freezing agents present in some electric home appliances forces them to be transported separately; this entails a subdivision of the equipment in each collection point. The assignment of a harvesting point to a territory is independent for each category, so that, it may be the case that a collection point is assigned to different territories for each category. Nevertheless, for reasons of efficiency the number of these cases is allowed only to certain limits.

In some countries such as Germany and Spain the normative is applied in a way to prevent monopolistic practices, this characteristic has motivated the use of a mathematical model that seeks to maximize a dispersion measurement. Therefore, the problem consists of creating territories that fulfill the different criteria of planning, and such that they are as disperse as possible.

Territorial design is already one of the classic problems within the field of operations research that, although widely studied for more than thirty years, has not been exhausted due to its versatility and capacity of adaptation to a rich variety of real situations. The nature of the problem just described locates it within the area of territorial design, but at the same time it displays char-

acteristics that make it relevant. The desire to obtain territories balanced with respect to diverse planning criteria turns it into a non-trivial problem. On the other hand, the need to maximize a dispersion measurement, gives it a special place as a unique problem within the field of territory design.

It is important to note that, in the review of specialized literature, except for the work carried out by Fernández et al. [4], there are no articles that approach a problem of maximum dispersion for territory design. This emphasizes the scientific contribution of this research. It is important to mention that the motivation of the work developed here, is indeed the investigation of these authors. In that work, the authors propose a constructive heuristic based on GRASP which contains a simple local search phase. The idea is to contribute with a more sophisticated local search.

In this paper we present a Tabu Search (TS) metaheuristic for this combinatorial optimization problem. The TS is further enhanced by a strategic oscillation component. Several neighborhoods and search strategies are developed as well. In addition, we implement an upper bounding scheme based on different relaxation strategies. Our experimental work fully assesses all different strategies and components of the proposed TS. The trade-offs between the different local search strategies are exposed. It was also observed the positive impact that strategic oscillation implementation has over all instances tested in terms of solution quality and number of proven optimal solutions found. The results indicated the overall efficiency of the algorithm, including significant improvements over the best solutions reported by a previously presented heuristic based on GRASP.

The rest of the paper is organized as follows. The problem is described in Section 2, including a combinatorial optimization formulation. Relevant work on this area is surveyed in Section 3. The proposed TS metaheuristic is fully described in Section 4. Section 5 presents the empirical work, assessing each of the algorithmic strategies and components. We wrap up in Section 6 with closing remarks and conclusions.

2 Problem Description

This problem was formally introduced by Fernández et al. [4]. In that work, the problem and modeling assumptions are fairly well described and motivated. In this work, we provide a summary of the main assumptions and planning requirements and present a combinatorial optimization model that will be used in the proposed solution procedure.

Let $V = \{1, \ldots, n\}$ be the set of basic units (BUs). In this case, a BU corresponds to a recollection point. Let w_i be the number of households of basic unit $i \in V$ and $W = \sum_{i \in V} w_i$ the sum of all households. Each BU is further classified according to the quality of its infrastructure. Denote by V_1, V_2, and V_3 the set of BUs of good, medium, and low quality, respectively. We use $q \in Q = \{1, 2, 3\}$ as an index for the respective quality sets and denote $q_i \in Q$ the quality logistics index of basic unit i. Under this definition, it is clear that $V_q = \{i \in V \ : \ q_i = q\}$, for $q \in Q$. White goods are further subdivided into devices that have freezing capabilities and those that do not (denoted as 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. Let d_{ij} be the

distance between BUs i and j, $i, j \in V$. We denote by $C = \{1, \ldots, m\}$ the set of corporations and M_k^p the market share of corporation $k \in C$ for product $p = 1, 2$. 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 may not be the same as the one that is responsible for the type 2 products. A BU whose company assignemnt is different for both product types is called a *split unit*.

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.

The following planning requirements are sought:

- For each type of product $p \in P$, a BUs must be asigned to a company, that is for each product p the assignment forms a p-partition of V.
- The numbre of split units is bounded by a user-specfied parameter σ.
- The total number of households should be fairly assigned to companies based on their market share for each product type.
- The number of recollection points of a specific quality index should be fairly assigned to companies based on their market share for each product type.

What makes the problem different and interesting is that the plan must also satisfy the WEEE Directive that establishes that regional monopolies must be avoided, that is, units allocated in smaller subregions should be assigned to different companies. As shown by Fernández et al. [4], this is accomplished by maximizing a dispersion measure (described below). This contrasts with previous work on territory design and districting where usually territory compactness is desired, that is, a dispersion measure is minimized.

Under the above assumptions, we present the following combinatorial optimization model version of the MILP model introduced by Fernández et al. [4]. This is called the *Maximun Dispersion Territory Design Problem* (MDTDP).

Sets

$V = \{1, \ldots, n\}$	Basic Units (BUs)
$C = \{1, \ldots, m\}$	Territories
$P = \{1, 2\}$	Product types
$Q = \{1, 2, 3\}$	Quality index (1=good, 2=medium, 3=low)
V^q	Set of BUs with quality $q \in Q$; $V = V^1 \cup V^2 \cup V^3$

Parameters

d_{ij} Euclidean Distance between BUs i and j; $i, j \in V$

w_i Number of households in BU i; $i \in V$

S_k^p Market share of territoy k for product p; $k \in C$, $p \in P$

τ Tolerance parameter with respect the number of households balance; $\tau \in (0, 1)$

β Tolerance parameter respect to BU quality; $\beta \in (0, 1)$

σ Maximum number of split BUs allowed

Computed Parameters

$w(\bar{V})$ $(= \sum_{i \in \bar{V}} w_i)$ Number of households in $\bar{V} \subset V$

W $(= w(V))$ Total of households in V

$c^q(\bar{V})$ $(= |\bar{V} \cap V^q|)$ Cardinality of $\bar{V}$ for quality index q; $\bar{V} \subset V$, $q \in Q$

Decision Sets

X_k^p Set of BUs assigned to territory k for product p; $k \in C$, $p \in P$

X_k $(= \bigcup_{p \in P} X_k^p)$ Set of UBs assigned to territory k for at least one product; $k \in C$

X^p $(= \{X_1^p, \ldots, X_m^p\})$ m-partitions of V for product p; $p \in P$

X^{split} Set of split BUs, $i \in X^{\text{split}} \Leftrightarrow \exists k_1, k_2 \in C, k_1 \neq k_2$, such that $i \in X_{k_1}^1 \wedge i \in X_{k_2}^2$

Π Set of all possible m-partition of the form $X = (X^1, X^2)$ for V

MDTDP Model

Find $|P|$ m-partitions of the form $X = (X^1, X^2) \in \Pi$, such that X optimize the model:

$$\max_{X\in\Pi}\min_{k\in C}\min_{i,j\in X_k}\{d_{ij}\} \tag{1}$$

$$\text{subject to:} \quad \frac{1}{W}w(X_k^p) \le (1+\tau)S_k^p \qquad k\in C, p\in P \tag{2}$$

$$\frac{1}{W}w(X_k^p) \ge (1-\tau)S_k^p \qquad k\in C, p\in P \tag{3}$$

$$\frac{1}{|V^q|}c^q(X_k^p) \le (1+\beta)S_k^p \qquad q\in Q, k\in C, p\in P \tag{4}$$

$$\frac{1}{|V^q|}c^q(X_k^p) \ge (1-\beta)S_k^p \qquad q\in Q, k\in C, p\in P \tag{5}$$

$$|X^{\text{split}}| \le \sigma \tag{6}$$

As stated before and shown in [4], the objective function (1) that seeks to maximize territory dispersion is compatible with avoiding regional monopolies. Constraints (2)-(3) assure that the number of households is fairly distributed to companies based on their market share. Due to the discrete nature of the problem, it is practically impossible to obtain a perfect balance. Therefore, this balance is achieved by introducing a tolerance parameter $\tau \in (0,1)$ that measures the deviation from a perfect measure given by WS_k^p. Similarly, constraints (4)-(5) assure that the good, medium and low quality BUs are fairly allocated to companies based on their market share too. To this end, a user-specified tolerance parameter $\beta \in (0,1)$ is introduced for achieving this balance. These two set of balancing constraints are referred to as the *household* and *infrastructure quality* balancing constraints. Note that tolerance values of $\tau = \beta = 0$ corresponds to a perfect blance. Finally, constraints (6) sets a limit on the number of split BUs allowed. In practice this is around 20 % of the total number of BUs.

Computational complexity: The MDTDP is NP-hard [4]. By making $C = \{1,2\}, \sigma = 0, S_k^p = 0.5, \tau = 0, \beta = 1$, and $d_{ij} = 1$, the Set Partiononiing Problem is polinomially reducible to the MDTDP. The state of the art says that tractable instances of this problem, that is, instances that can be solved exactly, have in the order of 20-30 BUS and 3-4 companies. Our target instances have 100-300 BUs and 3-7 companies.

We now provide an example to illustrate a typical solution or design for the MDTDP.

Example: Figure 1 shows a graphical representation of a solution to MDTDP, where the left- and right-half of each BU represents the assignment of product type 1 and 2, respectively. For instance, we can see that BU 2 has been assigned to company/territory 3 and 1 for products 1 and 2, respectively. BUs 4, 8, and 9 are split too, that is, $X^{\text{split}} = \{2,4,8,9\}$. The solution sets are given by: for product 1, $X_1^1 = \{5,7,9\}$, $X_2^1 = \{3,4,8,10\}$, and $X_3^1 = \{1,2,6\}$; and for product 2, $X_1^2 = \{2,4,5,7\}$, $X_2^2 = \{3,10\}$, and $X_3^2 = \{1,6,8,9\}$. From the company perspective, we have:

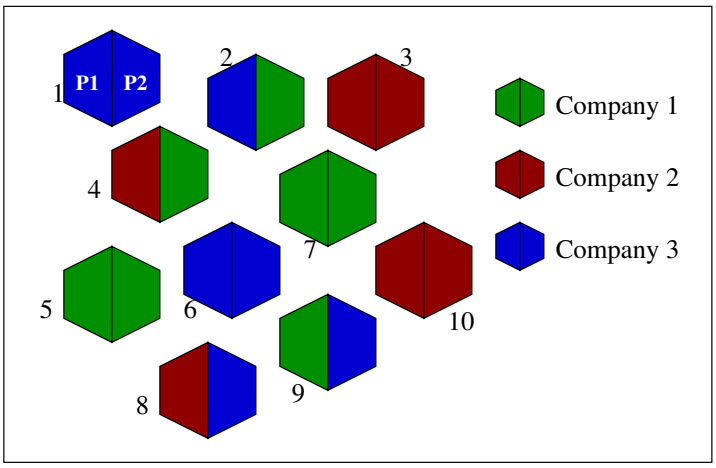

Figure 1: Graphical representation of a feasible solution to an MDTDP instance with 10 BUs, 3 territories, and 2 products.

$X_1 = \{2, 4, 5, 7, 9\}$, $X_2 = \{3, 4, 8, 10\}$, and $X_3 = \{1, 2, 6, 8, 9\}$

3 Related Work

As far as general districting and territory design problems are concerned, we refer the reader to the excellent surveys by Duque, Ramos, and Suriñach [2], Kalcsics [11], and Ricca, Scozzari, and Simeone [16]. In this section we review relevant works on WEEE recollection, and OR-related studies.

Although there are several articles referring to WEEE collection, most of them are of a qualitative nature. Walther and Spengler [20], for example conduct an analysis aiming at predicting what might be the impacts of new legal and economic developments on the treatment of discarded electronic products. Some articles center on technological issues. He et al. [9] review the implementation of strategies of WEEE treatment and the recovery technologies of WEEE in China, focusing on the attenuation of deteriorating effects of WEEE on the environment and the recovery of materials that can be reused in them. On the managerial aspects, Georgiadis and Besiou [5] provide some insight to the management of Closed Loop Supply Chains in order to attain either environmental and economical sustainability. Tsai and Hung [19] propose a two-stage multi-objective decision system, the first one involves the treatment of the collected material, in which a set of suppliers is selected, the second phase refers to recycling the recovered material. A study examining the two Swiss take-back and recycling systems one for computers, consumer electronics and telecommunication equipment, and one for household appliances in order to assess the environmental impact of recycling is presented by Hischier, Wäger, and Gauglhofer [10]. Their approach is based on material flow analysis and life cycle assessment, they conclude that WEEE recycling proves to be clearly advantageous from an environmental perspective when compared to incineration of all

WEEE and primary production of the raw materials Rudăreanu [17] explores the relationships among the agents that constitute waste management systems, the potential adverse health and environmental consequences of incorrect handling and treatment of WEEE, and the logistics of setting up and running a national WEEE management system. He discusses how his study impacts the development of the WEEE management system in Spain, the benefits of EEE to the society, and the potential effects of WEEE on health and environment. In a follow-up work, Rudăreanu [18] present a study on how the regulation of the WEEE II Directive impacts the WEEE management system in Romania.

There have been some technical papers from the OR perspective, Hammond and Beullens [8] model a network consisting of manufacturers and consumer markets engaged in a Cournot pricing game with perfect information, trying to attain equilibrium on volumes shipped and prices charged. Queiruga et al. [15] present a methodology based on PROMETHEE, an outranking method developed to solve problems of decision making under several objectives. The purpose of the decision is to select the most appropriate sites to locate WEEE recycling plants, and this methodology is applied to the case of locating recycling plants in Spain.

Mar-Ortiz, Adenso-Diaz, and González-Velarde [13] present a case study based on the design of a network for WEEE collection in the northern autonomic community of Galicia, Spain, although it is a case study they present methodological aspects of the design of reverse logistic networks including the vehicle routing problems that arise when such networks are configurated. In a second paper Mar-Ortiz, González-Velarde, and Adenso-Diaz [14] expand the afore mentioned VRP and introduce a new problem: vehicle routing with split loads and date windows, a characteristic that seems to be typical in reverse logistics problems. Lee and Shih [12] present a study that attempts to optimize end-of-life processes for electronic products based on a three-stage heuristic approach, which simultaneously minimizes cost and environmental impact. The proposed heuristic approach then assesses the most common disassembly and recycling processes by using the characteristics of electronic product recycling. Next, the best process for this bi-criteria optimization problem is identified by using the compromise programming method.

Fernández et al. [4] present a territory design where the territories should be as dispersed as possible, since each will be assigned to a logistics provider, and monopolies in the service are to be avoided. This seems to be the first and only article, to the best of our knowledge, in territory design motivated by the WEEE European directive. Our work is a follow-up of this work.

4 Proposed Tabu Search with Strategic Oscillation

Tabu search [7] is an iterative local search-based metaheuristic most commonly used in combinatorial optimization. Since its early inception Tabu Search (TS) has been successfully applied to a number of very hard combinatorial optimization problems in many fields, including districting problems [1]. Starting from an initial solution X^0, TS moves at each iteration t from a solution X^{t-1} to the best solution in its neighborhood $N(X^{t-1})$, even if this causes a deterioration in the value of the objective function. To prevent cycling, some solutions possessing particular attributes

are declared forbiden, or *tabu* for a given number of iterations. This number of iterations is refereed to as *tabu tenure*. The search stops whenever a stopping criterion is satisfied. The method can be improved through the incorporation of several features, some of which exploit the mathematical structure of the specific problem. We now describe in detail each component of the proposed TS for the MDTDP.

4.1 Neighborhoods and search strategies

The following three moves give rise to three different neigborhood structures.

- $move_{A1}(i,k)$: Reassign BU i from its current territory (denoted by $k(i)$) to a different territory k, with $k \neq k(i)$, for all the products.
- $move_{A2}(i,k,p)$: Reassign BU i from its current territory for product p (denoted by $k(i,p)$) to a different territory k, with $k(i,p) \neq k$.
- $move_{B}(i,j,p)$: Swap BUs i and j for product p, that is assign BU i for product p to territory $k(j,p)$ and assign BU j for product p to territory $k(i,p)$.

Let $N_{A1}(X)$, $N_{A2}(X)$, and $N_B(X)$, be the corresponding neighborhoods of design X formed by all solutions reachable from X by performing a move of type A1, A2, and B, respectively.

In the proposed method we explore two different search strategies made up of these neighborhoods. Neighborhood $N_{C1}(\cdot)$ or strategy C1 explores these neighborhoods sequentially as $N_{A1} \rightarrow N_{A2} \rightarrow N_B$. That is, the search is done over $N_{A1}(\cdot)$ for a given number of iterations, then $N_{A2}(\cdot)$ for a given number of iterations, and then $N_B(\cdot)$. A second strategy C2 is to consider a neighborhood made up of the union of the three, that is $N_{C2}(\cdot) = N_{A1}(\cdot) \cup N_{A2}(\cdot) \cup N_B(\cdot)$, which is explored through a given number of iterations. The stopping criteria is either local optimality or a fixed maximum number of iterations reached. In addition, a global optimality criterion that consists of a comparison with a previously computed dual bound is performed. In this last case, the algorithm stops with a proven global optimal solution. More about these dual bounds will be discussed in Section 4.7.

4.2 Recency-based memory and tabu tenure

Given the neighborhhods have polinomial size, a *best found* strategy is adopted, that is, the neighboorhood is entirely explored and the best non-tabu move is taken. To prevent cycling, whenever a move $move_{A1}(i,k)$ is performed we make BU i tabu so any move involving BU i is forbidden for θ iterations. Similarly, whenever a move $move_{A2}(i,k,p)$ is performed, BU i and territory k are declared tabu, and whenever a move $move_{B}(i,j,p)$ is performed, BUs i and j are declared tabu. A *dynamic tabu tenure* strategy is used, where every time θ is randomly drawn from the interval $[\theta_{\min}, \theta_{\max}]$. This idea practically removes the probability of cycling provided $\theta_{\min}$ and $\theta_{\max}$ are large enough. Fine-tuning of these limits is carrried out.

4.3 Merit function

It is common tu use a merit function to guide the search towards better solutions. In this case, since some of the constraints are being relaxed, the merit function is composed by the original objective function and some penalized terms in the objective function that measure the degree of insatisfaction of the relaxed constraints. The merit function maximized throughout the search is given by:

$$F(X) \quad = \quad \hat{f}(X) - \delta_\tau f_\tau(X) - \delta_\beta f_\beta(X) - \delta_\sigma f_\sigma(X), \tag{7}$$

where $\hat{f}(X)$ is the normalized objective function (1). The terms $f_r(X)$, with $r \in \{\tau, \beta, \sigma\}$ are functions that measure the degree of relative violation of the balancing constraints with respect to the number of households (2)-(3), balancing constraints with respect to the infrastructure quality (4)-(5), and maximum number of split units allowed (6), respectively. The parameters δ_r are self-adjusted multipliers. These multipliers are initially set to 1 and allowed to vary during the search to account for the fact that any given solution X may be infeasible with respect to any of these constraints. This is based in the concept of strategic oscillation and are further explained in Section 4.4.

4.4 Strategic oscillation

The idea behind *strategic oscillation* [6] is to guide the search through both the feasible and infeasible space to gain more flexibility and reach portions of the solution space that would be impossible to explore otherwise. To this end, some of the constraints are relaxed and moved into the objective function with a self-adjustable penalty parameter. It is the self-adjustment mechanism of these penalty parameters what allows to guide the search between the feasible and infeasible space. This technique has proven successful in many combinatorial optimization problems, particularly in some territory design applications. For instance, Bozkaya, Erkut, and Laporte [1] make use of this idea for successfully handling some difficult constraints in a political districting problem. In our case, the parameters λ_r in the merit function are initially set to $\bar{\delta}_r$, and adjusted every μ_r iterations according to the following rule: If all previous $\bar{\mu}_r$ solutions were infeasible then $\delta_r = \gamma\,\delta_r$; if all of them were feasible then $\delta_r = \frac{1}{\gamma}\,\delta_r$; else δ_r remains unchanged. The parameters μ_r, and $\bar{\mu}_r$ are positive integers and $\gamma > 1.0$ is a real number. These are user-controlled fixed parameters.

4.5 Aspiration criterion

The following aspiration criterion is incorporated into the TS procedure. If the objective function value of the best neighbor found is better than the objective function value of the best solution found so far, then the move is taken even if it is a tabu move.

4.6 Summary of the Proposed Tabu Search Algorithm

The proposed Tabu Search for the MDTDP (called TS_MDTDP) is depicted in Pseudocode 1. X^{best}represents the best solution found so far and t is the iteration counter. $T(t)$ is the set of tabu moves associated to iteration t and A is the set of solutions that satisfy the aspiration criterion for the incumbent solution. The initial solution $\hat{X}_0$ can be obtained by any of the construction procedures described in Fernández et al. [4]. In this case, we use construction procedure H1. In the process of choosing the best neighbor, $\aleph(\cdot)$ represents any of the previously described neighborhoods, whereas $\tilde{X}$ satisfies $F(\tilde{X}) > F(Y)$ for all $Y \in \{\aleph(\hat{X}_t) \setminus T(t)\} \cup A$.

A solution X_i is considered to improve X^{best}in any of the following cases:

- If feasibility has not been reached, X_i improves X^{best}, if X_i decreases its value of total relative infeasibility.
- Once feasibility has been achieved, X_i improves X^{best}, if $f(X_i) > f(X^{\text{best}})$ and X_i is feasible.

The local search continues until any of the following stopping criteria is met: (i) maximum relative optimality gap with respect to a dual (upper) bound; (ii) maximum number of iterations; (iii) maximum time limit. When stopping, the algorithm returns X^{best}, the best solution found. Note that, if the maximum relative optimality gap in (i) is set to zero, and this criterion is met when stopping, $X^{\text{best}}$is a global optimum.

Procedure 1 TS_MDTDP( )

Input: An instance to the MDTDP

Output: : X^{best}, A solution for the MDTDP

1: $t \leftarrow 0$
2: $T(0) \leftarrow \emptyset$
3: $\delta_r \leftarrow \overline{\delta}_r$, $r \in \{\tau, \beta, \sigma\}$
4: Obtain initial solution $\hat{X}_0$
5: $X^{\text{best}} \leftarrow \hat{X}_0$
6: **while** (stopping criteria not met) **do**
7: $t \leftarrow t + 1$
8: **if** ($t \mod \mu_r = 0$) **then**
9: Update δ_r
10: **end if**
11: Choose the best neighbor $\tilde{X} \in \{\aleph(\hat{X}_t) \setminus T(t)\} \cup A$
12: $\hat{X}_t \leftarrow \tilde{X}$
13: Randomly choose $\theta \in [\theta_{\min}, \theta_{\max}]$
14: Update $T(t)$
15: **if** ($\hat{X}_t$ is better than X^{best}) **then**
16: $X^{\text{best}} \leftarrow \hat{X}_t$
17: **end if**
18: **end while**
19: **return** X^{best}

4.7 Upper Bounding Schemes

The importance of dual bounds for optimization problems is well established. Among the benefits of having a dual bound we have:

- It allows to compute relative optimality gaps of feasible solutions so we can measure the quality of heuristic or primal solutions.
- As a consequence, global optimality can be proven if both primal and dual bounds are equal.
- A dual bound can sometimes be further improved by embbeding it within enumeration schemes such as branch and bound or dynamic programming.

For this problem, it is possible to compute upper (dual) bounds with a relatively short computational effort. Note first that if we relax constraints (6), the remaining (relaxed) problem is simply a partitioning problem consisting of finding a node partition such that the given measure for dispersion is maximized. We refer to this problem as the *Unconstrained Maximum Dispersion*

Problem (UMDP). Clearly, any valid relaxation or upper bound for UMDP is also valid for our MDTDP.

In [3], Fernandez et al. developed an upper bound for the UMDP. This is roughly based on the following idea. For any arbitrary subset of cardinality $m+1$ BUs, at least two of the BUs must belong to the same territory. Let i and j be these two BUs from set X belonging to the same territory. Clearly, among all possible combinations, the worst case occurs when i and j are as far away from each other as possible. Therefore, $\text{UB}(X) = \max_{i,j \in X}\{d_{ij}\}$ is a valid upper bound for the optimal solution to UMDP. Now, there exists $\binom{n}{m+1} = \frac{n!}{(m+1)!(n-(m+1))!}$ different ways of chosing subsets of size $m+1$ from a set of size n. The main issue in computing the upper bound is to choose a subset X of $m+1$ BUs in such a way that X gives the best (lowest) possible value of the upper bound. Since it is not practical to generate all possible subsets X of size $m+1$, the idea is just to obtain an approximation by a smart choice of a few of those subsets. It can be easily seen that if $\mathcal{X}$ is a collection of subsets of V of cardinality $m+1$ each, then $\max_{X \in \mathcal{X}}\{\text{UB}(X)\}$ is the best possible upper bound on the optimal value of UMDP among the sets of this collection. The authors propose a simple heuristic to obtain an attractive collection of subsets, by iterating over each BU i and then form its associated subset of size $m+1$ by choosing the m nearest BUs to i. By iterating over each BU i, a collection of n subsets is formed and an upper bound is computed. See Pseudocode 2.

Procedure 2 UB_FKN( )

Input: An instance to the MDTDP

Output: UB^{best}: A valid upper bound for the MDTDP problem

1: $\text{UB}^{\text{best}} \leftarrow \infty$
2: **for** ($i = 1, \ldots, n$) **do**
3: $X \leftarrow \{i\}$
4: $X \leftarrow X \cup \{m \text{ closests nodes to } i\}$
5: $bound \leftarrow \max_{j,k \in X}\{d_{jk}\}$
6: **if** ($bound < \text{UB}^{\text{best}}$) **then**
7: $\text{UB}^{\text{best}} \leftarrow bound$
8: **end if**
9: **end for**
10: **return** UB^{best}

Now, based on this idea and recognizing the fact that there might be different ways of choosing BUs "close to" BU i (Step 4 in Pseudocode 2), we propose three different strategies, each yieding a different collection, and therefore a different bound.

UB1: First, rather that using the complete distance matrix D, we use a truncated matrix consisting of the $2m$ closest units to each node, that is, $\bar{D}_{ij}$ is a matriz of dimension $(n \times 2m+1)$ where each row i contains the $2m$ lowest values of d_{ij} for all $j = 1, \ldots, n$. Let A_i the set of these

$2m$ BUs closest to i. The other important difference with respect to Pseudocode 2 is the computation of Step 4. Rather than choosing the m closest units, we perform this procedure iteratively, one unit at a time, taking into account not only node i but the nodes that have already been added to set X as follows. Suppose that for a fixed unit i and a partial set X being formed in Step 4, we compute the "distance" of each unit $q \in A_i \setminus X$ to set X as $d(X,q) = \max_{j,k \in X \cup \{q\}} \{d_{jk}\}$. This distance estimates the value of the upper bound if unit q were to be added to set X. Then, unit $q^* = \arg\min_{q \in A_i \setminus X} \{d(X,q)\}$ is chosen and added to X. We proceed this way until a set X of size $m+1$ is formed. The rest of the procedure remains the same.

Figures 2 illustrate the selection process. In this example $X = \{i, j\}$ and $A_i \setminus X = \{k, r\}$. According to the distances shown in the figures, $d(X,k) = \max\{d_{ij}, d_{ik}, d_{jk}\} = \max\{1.9, 1.7, 4.2\} = 4.2$ and $d(X,r) = \max\{d_{ij}, d_{ir}, d_{jr}\} = \max\{1.9, 2.0, 2.3\} = 2.3$. Therefore unit r is chosen and added to X.

Figure 2: Computation of $d(X,k)$ and $d(X,r)$.

UB2: This is exactly the same procedure as UB1 with the exception than the original distance matrix D is used instead of the truncated matrix $\bar{D}$.

UB3: The idea behind UB3 is to attempt to avoid repetition of subsets that may occur when UB2 is applied. For instance, at iteration j, we fixed unit j in set X and then its first unit to be added to set X is unit i. However, if it turns out that $i < j$ and in a previous iteration j was the first unit added to set $X = \{i\}$, the rest of the procedure will choose the same subset. To avoid this, we design UB3 that it is basically UB2, but restricting the choosing of the first element to be added to set $X = \{j\}$, by a lexicographic rule, to only $k > j$. The rest of the procedure remains the same.

Now, each of these three procedures runs in $O(n)$ and produces a different bound. Thus, to have the best possible bound, we basically apply all three procedures to build potentially $3n$ subsets and then take the best bound among all $3n$ subsets. This is done very quickly.

Upper Bound Computations

We now provide some preliminar computational testing on the upper bounding schemes. For this purpose we use the 96-instance database (fully described in Section 5).

Table 1: Comparison of upper bounding schemes.

	Gap		
n	UB1	UB2	UB3
100	0.36	0.35	0.36
150	0.40	0.40	0.41
200	0.37	0.37	0.37
250	0.15	0.16	0.16
300	0.07	0.07	0.07
Average	0.27	0.27	0.27

Table 1 shows the relative optimality obtained by each upper bounding scheme as a function of the number of BUs. This optimality gap is computed by using the best known upper bound at this point (the one obtained by the GRASP of Fernández et al. [4]). As can be seen, the best results are obtained for the larger instances, obtaining average gaps of around 16% and 7% for the 250 and 300/BU instances, respectively. the results for the 100- to 200-BU instances. A typical behavior of GRASP is that many times the best results are reached for larger size instances, thus this might explain the behavior in the smaller instances.

Table 2: Running times for the upper bounding schemes.

	Average time (sec.)		
n	UB1	UB2	UB3
100	0.01	0.01	0.01
150	0.01	0.02	0.02
200	0.02	0.03	0.02
250	0.03	0.04	0.04
300	0.04	0.04	0.06
Minimim	0.01	0.01	0.01
Average	0.02	0.02	0.03
Maximum	0.05	0.06	0.08

Table 2 display the average running times for each upper bounding scheme. As can be seen, the procedures run rather quickly even for the largest instances. These are average times, but it was found that no instance required more than 0.08 seconds of running time.

Non-parametric tests (Wilcoxon and Mann-Whitney) were applied confirming no statistical difference exists among the behavior of the upper bounding schemes in terms of their quality. Due to this and to their relatively low computational cost, it was decided that the best strategy for

obtaining the best possible bound was to apply all three procedures, and take the best out all of these, that is, UB = min { UB1, UB2, UB3 }. This is computed only once as a pre-processing phase prior to the Tabu Search, and is used as stopping criterion, in addition to the number of iterations.

Finally, Figure 3 shows a comparison between the UB computed this was and the best known solution at this point (obtained by GRASP) for the 250- and 300-BU instances.

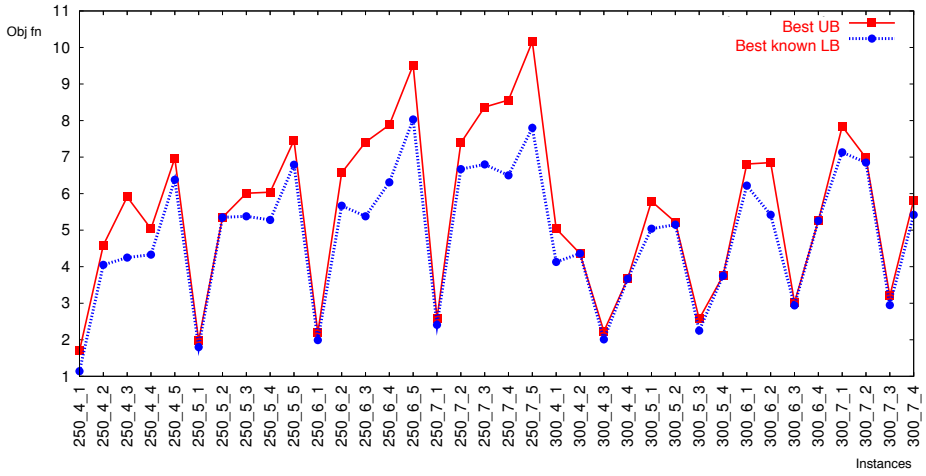

Figure 3: Comparison between best UB and LB obtained by GRASP [4] for 250- and 300-BU instances.

5 Computational experiments

For the experiments, we used the data instances taken from Fernández et al. [4]. We build initial solutions by using heuristic H1 from that work with parameters $\alpha = 0.2$ and $\lambda = 0.5$. The size of these instances range from 100 to 300 BUs and 4 to 7 territories. For each of the 100- to 250-BU instances there are 5 instances per size and for the 300-BU instances there 4 instances per size, for a total of 96 tests instances.

The following notation is used to identify each instance: n_p_x, where n, p and x denote number of BUs, number of territories, and instance ID number. For example, sufix 150_4_3 represents the third instance with 150 BUs and 4 territories. Symbol (†) indicates that the best reported solution did not reach feasibility, while symbol (*) denotes an optimal solution.

The relative optimality gap, or just gap, is computed as:

$$\text{gap} = (\text{UB} - \text{LB})/\text{LB},$$

where UB is the best found upper bound computed by the upper bounding schemes described in Section 4.7, and LB is the heuristic solution by the corresponding heuristic.

Unless otherwise noticed, all experiments were carried out using $\tau = 0.05$, $\beta = 0.20$ and

$\sigma = 0.20n$. The following Tabu Search algorithmic parameters were found to give the best results in preliminary testing and are therefore used throughout the experimentation: $\theta_{\max}$ =15, $\theta_{\min}$ =5, $\gamma = 1.5$, $\mu_r = 10$, $\overline{\mu}_r = 3$, 3000 move iteration limit for neighborhood $\aleph_{C2}$, 1000 move iteration limit for $\aleph_{C1}$ (for each of ($\aleph_{A1}$, $\aleph_{A2}$, and $\aleph_B$), that are sequentially explored.)

All procedures and methods of the Tabu Search were coded in C++ and compiled with the GNU C++ compiler (g++) under Ubunto 9.05 OS. A Gateway workstation with Intel Core 2 Duo T6400 processor at 2.0 GHz with 4 GB of memory was used.

5.1 Experiment A: Neighborhood Assessment

The goal of this experiment is to compare the performance of neighborhoods $\aleph_{C1}$ y $\aleph_{C2}$, decribed in Section 4.1, within the TS scheme. It is important to note that for each of the neighborhoods the TS found feasible instances in 100% of the instances tested. In each of the figures, $\aleph_{C1}$ and $\aleph_{C2}$ are identified by C1 and C2, respectively.

Table 3: Comparison between $\aleph_{C1}$ and $\aleph_{C2}$.

	$\aleph_{C1}$	$\aleph_{C2}$
Gap (average)	0.30	0.23
Time - average (sec.)	326.15	648.34
Time - minimum (sec.)	4.26	35.92
Time - maximum (sec.)	1830.63	2413.51
Number of optimal solutions found	7	7

Table 3 presents a summary of the average results for 96 instances tested. Figures 4 and 5 show the behavior of objective function and running time, respectively, for all instances with 100 and 150 BUs. Figures 6 and 7 show the behavior of objective function and running time, respectively, for the larger instances (200 to 300 BUs.)

As can be seen, the average running time for $\aleph_{C2}$ is almost twice as much that the one employed by $\aleph_{C1}$. The lowest average relative optimality gap was obtained under $\aleph_{C1}$, though. Given the overall running times are not too high (around 10 minutes per instance), one could afford to use $\aleph_{C2}$ in future experiments.

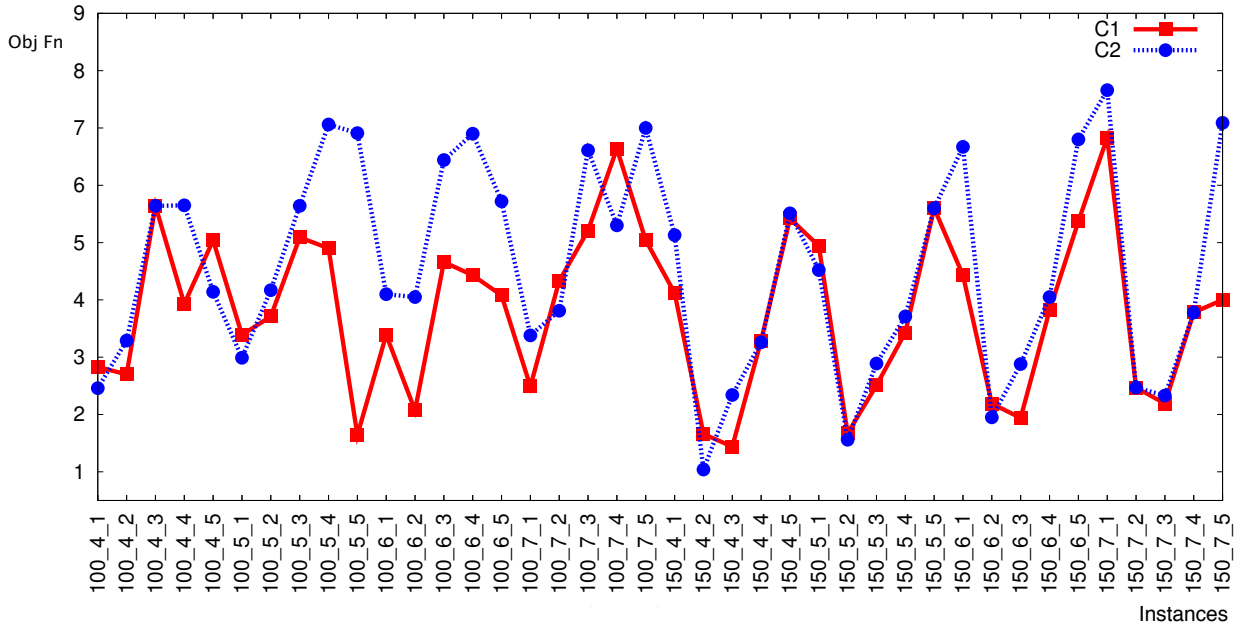

Figure 4: Objective function comparison between $\aleph_{C1}$ and $\aleph_{C2}$ on 100- and 150-BU instances.

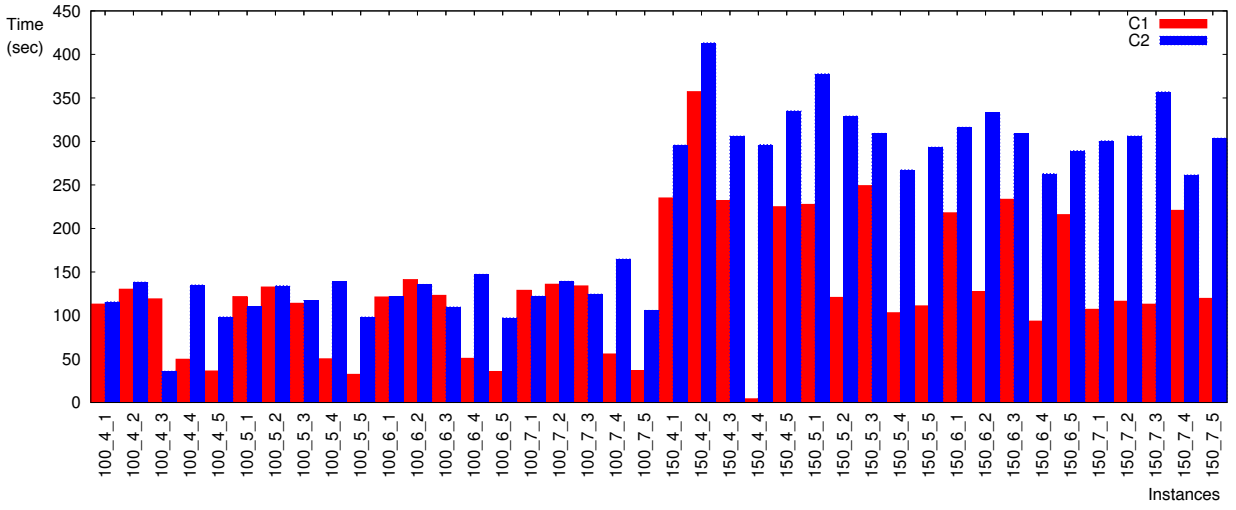

Figure 5: Running time comparison between $\aleph_{C1}$ and $\aleph_{C2}$ on 100- and 150-BU instances.

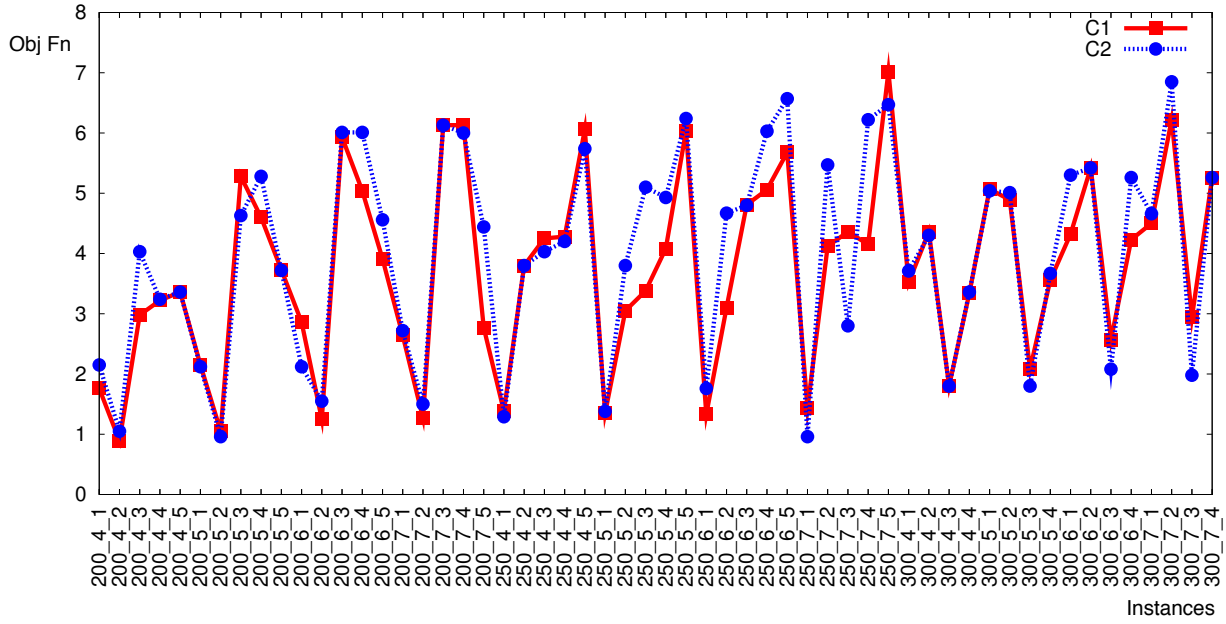

Figure 6: Objective function comparison between $\aleph_{C1}$ and $\aleph_{C2}$ on 200- to 300-BU instances.

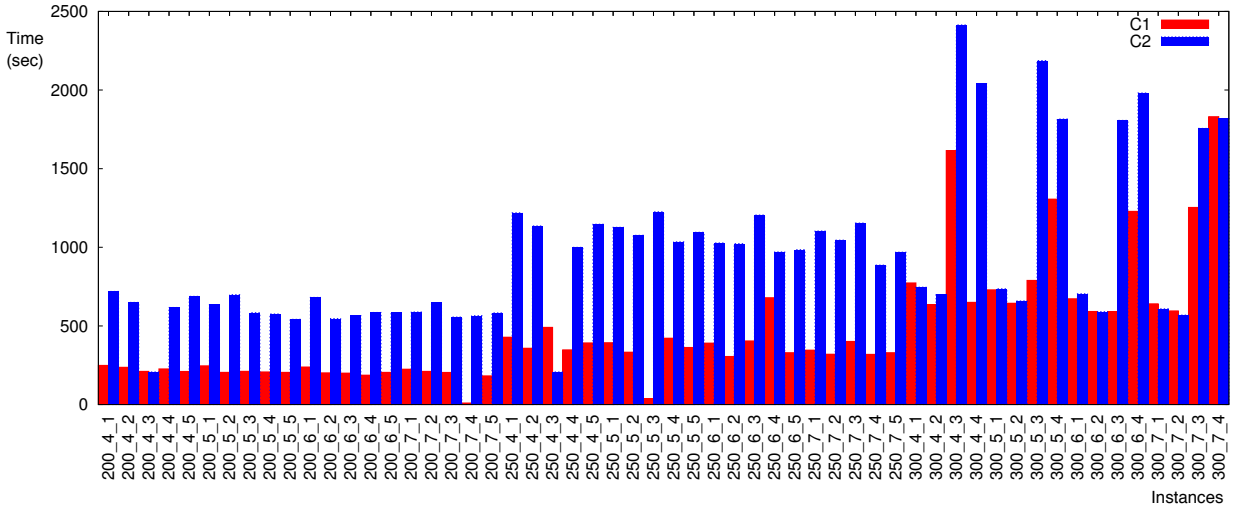

Figure 7: Running time comparison between $\aleph_{C1}$ and $\aleph_{C2}$ on 200- to 300-BU instances.

5.2 Experiment B: Assessment of Strategic Oscillation

As stated before, one of the advanced features implemented in our algorithm is that of strategic oscillation. The goal of this experiment is to assess the benefit if this component within a TS algorithmic framework. To this end, we applied the TS to all instances under two different strategies following the neighborhood $\aleph(\cdot) = \aleph_{C1}$. We run TS without the strategic oscillation component (denoted by index NSO) and then run the TS with the strategic oscillation component (denoted by subindex SO).

Tables 4-8 display all the results for each individual instance. The value of the objective function is shown in columns X_{NSO} and X_{SO}. The relative optimality gap is shown in the "gap" columns. The term "na" indicates no gap was found because no feasible solution (lower bound) was found. Time (time) is shown in CPU seconds. Table 9 summarized these results over all 96 instances in terms of average relative optimality gap, running time, number of infeasible solutions delivered, and number of optimal solutions found.

Table 4: Assessment of strategic oscillation for 100-BU instances.

Instance	X_{NSO}	gap	time	X_{SO}	gap	time
100_4_1	1.99	0.56	41.64	3.39	0.25	40.78
100_4_2	3.12	0.20	43.67	3.26	0.17	43.81
100_4_3	5.64*	0.00	2.14	5.64*	0.00	37.63
100_4_4	4.33	0.47	49.53	4.81	0.41	50.76
100_4_5	3.87	0.39	33.24	5.04	0.20	36.04
100_5_1	2.99	0.47	38.92	3.39	0.40	37.49
100_5_2	2.40	0.45	43.08	3.71	0.15	44.07
100_5_3	5.09	0.11	39.02	5.09	0.11	36.71
100_5_4	4.12	0.57	47.83	5.30	0.45	49.42
100_5_5	2.30	0.67	35.31	2.22	0.68	33.90
100_6_1	4.13	0.38	41.01	4.11	0.39	39.39
100_6_2	2.07	0.57	47.18	2.70	0.45	45.61
100_6_3	4.73	0.34	39.65	4.66	0.35	43.64
100_6_4	5.76	0.47	49.91	6.22	0.43	51.62
100_6_5	5.83	0.26	33.83	5.83	0.26	35.44
100_7_1	2.87	0.61	38.77	2.83	0.62	41.66
100_7_2	4.44†	na	48.22	4.99	0.15	48.37
100_7_3	6.61	0.08	40.85	5.85	0.19	41.86
100_7_4	5.57	0.53	50.99	6.64	0.44	53.68
100_7_5	2.07†	na	33.00	5.04	0.48	36.59

Table 5: Assessment of strategic oscillation for 150-BU instances.

Instance	X_{NSO}	gap	time	X_{SO}	gap	time
150_4_1	3.49	0.49	113.37	5.12	0.25	106.86
150_4_2	1.56	0.07	115.71	1.67*	0.00	6.93
150_4_3	1.94	0.21	112.01	1.94	0.21	111.58
150_4_4	2.85	0.13	102.9	3.29*	0.00	101.22
150_4_5	5.32	0.26	110.49	5.51	0.24	108.91
150_5_1	4.55	0.51	104.70	6.38	0.31	110.78
150_5_2	1.64	0.22	116.62	1.94	0.08	123.89
150_5_3	2.65	0.15	112.81	2.51	0.19	113.51
150_5_4	3.70	0.15	101.25	3.43	0.21	106.16
150_5_5	5.65	0.30	108.41	5.76	0.29	107.66
150_6_1	3.89	0.65	104.03	4.50	0.60	103.27
150_6_2	1.5	0.32	117.36	2.19*	0.00	119.75
150_6_3	2.22	0.34	106.04	2.13	0.36	107.97
150_6_4	3.43	0.30	90.12	3.82	0.22	101.16
150_6_5	4.52	0.53	102.86	6.15	0.36	110.41
150_7_1	5.35	0.55	100.37	6.86	0.43	101.90
150_7_2	2.47	0.26	112.62	2.47	0.26	106.57
150_7_3	1.74	0.55	102.88	2.19	0.44	112.40
150_7_4	3.54	0.40	95.12	3.78	0.35	96.43
150_7_5	4.34	0.63	110.54	6.45	0.45	107.04

Table 6: Assessment of strategic oscillation for 200-BU instances.

Instance	X_{NSO}	gap	time	X_{SO}	gap	time
200_4_1	2.15	0.21	240.62	2.15	0.21	247.30
200_4_2	1.05	0.22	227.04	1.05	0.22	214.59
200_4_3	2.23	0.53	211.21	3.24	0.31	222.22
200_4_4	3.02	0.36	222.42	3.24	0.31	203.92
200_4_5	3.26	0.13	203.13	3.36	0.10	209.54
200_5_1	2.12	0.31	241.47	2.72	0.11	242.91
200_5_2	1.23	0.28	210.85	1.14	0.34	221.11
200_5_3	4.63	0.12	198.16	5.28*	0.00	193.45
200_5_4	5.28*	0.00	127.84	4.80	0.09	188.66
200_5_5	3.73	0.17	204.59	3.73	0.17	201.81
200_6_1	2.17	0.42	242.38	2.86	0.23	225.82
200_6_2	1.32	0.35	195.71	1.32	0.35	191.37
200_6_3	5.28	0.12	192.85	5.93	0.01	191.55
200_6_4	5.93	0.01	192.06	5.93	0.01	189.37
200_6_5	2.72	0.46	215.06	3.91	0.23	198.82
200_7_1	2.72	0.36	237.97	2.72	0.36	239.35
200_7_2	1.24	0.42	206.59	1.27	0.41	205.36
200_7_3	6.13*	0.00	10.26	6.13*	0.00	206.16
200_7_4	6.13*	0.00	10.10	6.13*	0.00	10.16
200_7_5	4.33	0.24	197.88	3.75	0.34	189.50

Table 7: Assessment of strategic oscillation for 250-BU instances.

Instance	X_{NSO}	gap	time	X_{SO}	gap	time
250_4_1	1.33	0.22	434.28	1.38	0.19	394.01
250_4_2	3.22	0.30	365.25	3.80	0.17	358.56
250_4_3	3.79	0.36	453.12	4.25	0.28	478.03
250_4_4	4.14	0.18	355.40	4.28	0.15	357.42
250_4_5	4.40	0.37	368.60	6.06	0.13	375.46
250_5_1	1.45	0.27	407.72	1.70	0.14	343.95
250_5_2	3.40	0.36	338.03	3.26	0.39	334.76
250_5_3	4.99	0.17	447.32	4.99	0.17	431.55
250_5_4	3.38	0.44	327.83	4.08	0.32	340.29
250_5_5	6.19	0.17	369.10	6.19	0.17	352.66
250_6_1	1.10	0.50	378.45	1.51	0.31	364.99
250_6_2	3.05	0.54	312.04	3.11	0.53	317.46
250_6_3	4.36	0.41	403.25	4.81	0.35	411.79
250_6_4	4.90	0.38	319.47	5.27	0.33	322.72
250_6_5	6.91	0.27	327.66	6.91	0.27	319.48
250_7_1	1.89	0.27	337.03	1.89	0.27	342.87
250_7_2	5.74	0.22	297.54	5.43	0.27	293.90
250_7_3	4.40	0.47	377.00	5.45	0.35	396.10
250_7_4	5.55	0.35	327.08	6.20	0.28	312.16
250_7_5	5.71	0.44	342.11	7.04	0.31	333.55

Table 8: Assessment of strategic oscillation for 300-BU instances.

Instance	X_{NSO}	gap	time	X_{SO}	gap	time
300_4_1	3.43	0.32	762.94	4.09	0.19	764.95
300_4_2	4.3	0.01	645.65	4.36*	0.00	660.01
300_4_3	2.01	0.10	696.44	2.01	0.10	695.31
300_4_4	3.34	0.09	645.40	3.13	0.15	659.67
300_5_1	5.06	0.13	746.41	5.07	0.12	741.49
300_5_2	3.88	0.26	623.45	5.01	0.04	628.96
300_5_3	1.61	0.37	700.35	2.08	0.19	711.23
300_5_4	3.49	0.07	596.42	3.56	0.05	619.00
300_6_1	4.46	0.35	624.03	4.32	0.37	745.56
300_6_2	5.42	0.21	589.07	5.42	0.21	583.54
300_6_3	2.23	0.26	610.36	2.57	0.15	647.34
300_6_4	4.25	0.19	573.51	4.25	0.19	579.05
300_7_1	4.68	0.40	582.63	4.50	0.43	567.60
300_7_2	4.10	0.41	553.07	6.27	0.10	571.73
300_7_3	2.90	0.09	594.38	2.94	0.08	632.68
300_7_4	4.69	0.19	583.93	5.26	0.10	598.92

An important observation is that, under SO, all solutions found were feasible, while under NSO, the TS failed in achieving feasibility in two instances. It can also be seen that the running times of either stragey are around the same. One could expect that SO would take longer to run; however, most of the runs stopped by iteration limit, so this explains the similarity on running times.

Table 9: Summary of strategic oscillation assessment.

	gap		time	
n	X_{NSO}	X_{SO}	X_{NSO}	X_{SO}
100	0.40	0.33	39.89	42.42
150	0.35	0.26	107.01	103.22
200	0.24	0.19	189.41	199.65
250	0.33	0.27	364.41	359.09
300	0.22	0.15	633.00	650.44
Average	0.31	0.24	251.48	255.15
Number of optimal solutions found	4	8		
Number of infeasible solutions	2	0		

As can be seen the use of SO yields a significant benefit in terms of solution quality. For each groups of instances, the SO produced significant improvements in relative optimality gap. In addition to this, under SO more proven optimal solutions were found (twice as much). This clearly demonstrates the tremendous benefit of SO in this particular problem.

5.3 Experiment C: Impact of Tabu Search over Simple Local Search

One should expect that TS performs better than a simple local search. The purpose of this experiment is to measure how much is gained by employing TS in terms of solution quality and feasibility concerns. In other words, what is the impact that TS brings to the table.

Table 10: Asessment of benefit of TS over LS.

	gap	
n	LS	TS
100	0.38	0.26
150	0.45	0.26
200	0.32	0.19
250	0.39	0.29
300	0.40	0.17
Average gap	0.39	0.23
Number of optimal solutions found	3	7
Number of infeasible solutions	5	0

To this end, we apply the TS (using the SO strategy) and a simple local search, that is, applying the same neighborhood in a hill-climbing manner, stopping when no better neighbor is found. The

results are displayed in Table 10, where LS and TS indicate the average relative optimality gaps found under the Local Search scheme and Tabu Search, respectively.

As we can see, the benefit of using TS is clear in all aspects. The quality of the solutions was significantly better when TS was employed achieving an average relative improvement of over 40%. This difference is even more dramatic for the largest set of instances. It was also observed that regular LS could not find feasible solutions to five instances, wheras TS was able to find feasibe solutions for all instances tested. Finally, TS was able to find more proven optimal solutions that the ones found by LS.

5.4 Experiment D: Tabu Search Performance

For this last experiment, the goal is to asesss the benefit of the proposed TS when compared to the best heuristic from literature, the GRASP of Fernández et al. [4]. Now, it is important to notice that GRASP is in essence a construction heuristic that builds a solution from scratch, whereas TS is a local search heuristic that takes a built solution as an input. Thus, the issue we want to investigate is the degree of improvement (if any) of the TS local search heuristic over the best solution built by GRASP. To this end, we first apply the GRASP with parameters $\alpha = 0.2$, $\lambda = 0.5$, 2000 iterations, heuristic H1 as constructive mechanism and local search LS2. We register the best solution found by GRASP in every instance. It was first observed that GRASP obtained proven optimal solutions to 9 of the 96 instances. Then, for the remaining 87 instances, we apply the TS taking the best GRASP solution as input. It was observed that in 47 instances out of 87, TS was able to improve the solution quality.

Figures 8 and 9 show a comparison between the initial solution (X^{ini}) fed to the TS, and the final solution found by TS (X^{best}) for 100- to 150-BU and 200- to 300-BU instances, respectively.

Table 11: Assessment of TS.

	gap		Improvement
n	X^{ini}	X^{best}	(%)
100	0.81	0.41	30
150	0.94	0.46	33
200	0.96	0.33	49
250	0.20	0.20	< 1
300	0.10	0.10	< 1
Average	0.64	0.31	24

Table 11 presents a summary of the results, where the second and third columns show the average relative optimality gaps computed at the start and end of the TS algorithm. The last column, displays the average relative improvement (%) obtained by TS, computed as: ARI $= (X^{\text{best}} - X^{\text{ini}})/X^{\text{ini}}$.

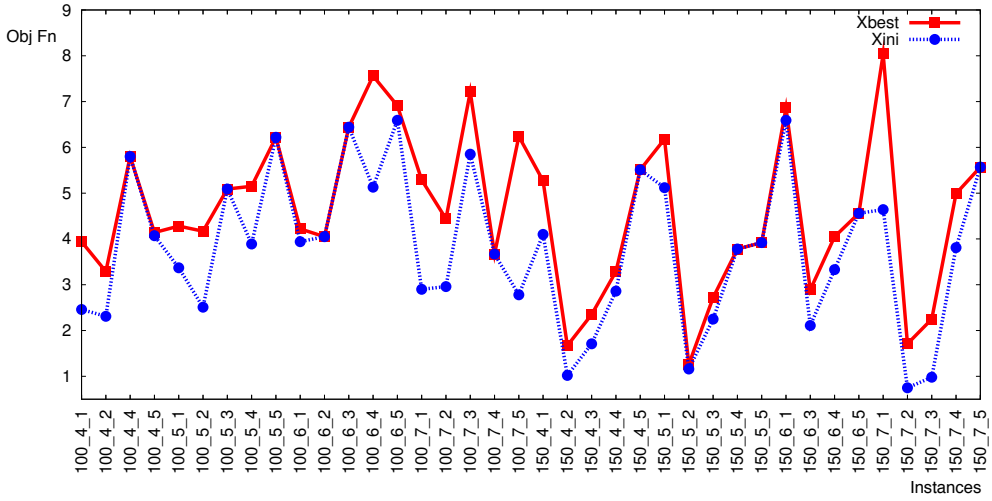

Figure 8: Comparison betweek the initial solution (X^{ini}) and the TS solution X^{best} for 100- to 150-BU instances.

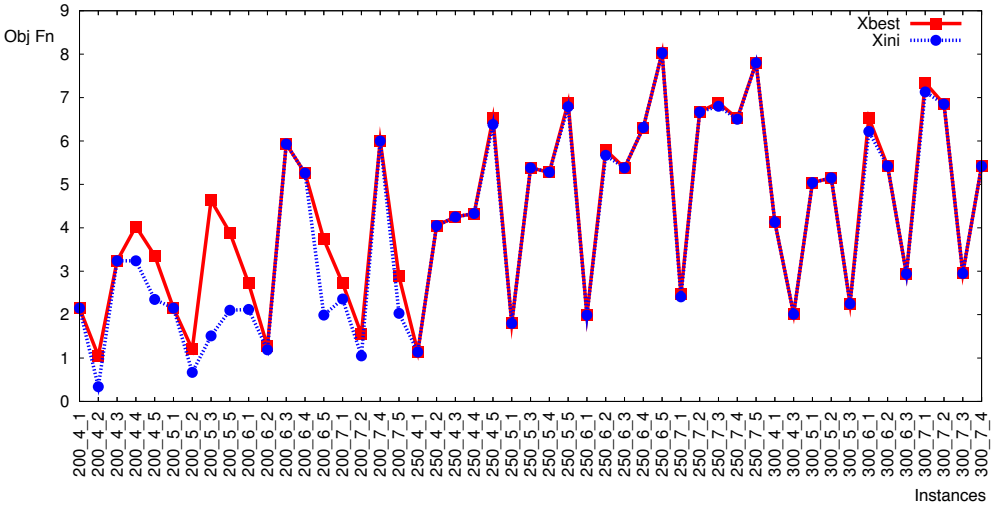

Figure 9: Comparison between the initial solution (X^{ini}) and the TS solution X^{best} for 200- to 300-BU instances.

It was clear from the start that a decrease in the relative optimality gap after applying TS was expected; however, it can be observed that this improvemente was substantial, particularly for the 100- to 200-BU instances. An explanation of this is that it is well known that in many combinatorial optimization problems, GRASP tends to perform better as the size of the instance grows. In these cases, the capacity of the TS to further improve the initial solution is more limited.

Finally, Figure 10 shows the behavior of the TS and its strategic oscillation component in a single instance of size 300 BUs and 4 territories. Objective function value versus move iteration within the TS is plotted. A circle denotes a feasible solution and a triangle denotes an infeasible solution. As we can see, the TS starts with an infeasible solution and objective function value of

around 2.83. Then, as the search goes on, a move to a worst objective (and still infeasible solution) is made in iteration 2. Through iteration 11 the solution remains infeasible. Then, a feasible solution is found (for the first time) in iteration 12 (with objective function value of around 2.79.) Then the trajectory moves at iteration 15 to a worst solution (value of around 2.22) and remains the same until iteration 27. At iteration 28, a better solution is found (but no better than 2.79), it keeps going up and down for 6 more iterations. At iteration 35, a solution with a better objective function value is found (at around 2.89); however, this is infeasible. Then the process finds a better infeasible solution and then recovers feasibility at iteration 37 with an objective of around 2.96 (the best solution so far). Thus, the plot beautifully illustrates both the typical TS behavior of going to a worse solution before improving again, and the strategic oscillation of moving between infeasible and feasible solutions. At the end, the best solution was found at iteration 70 with a value of 2.99.

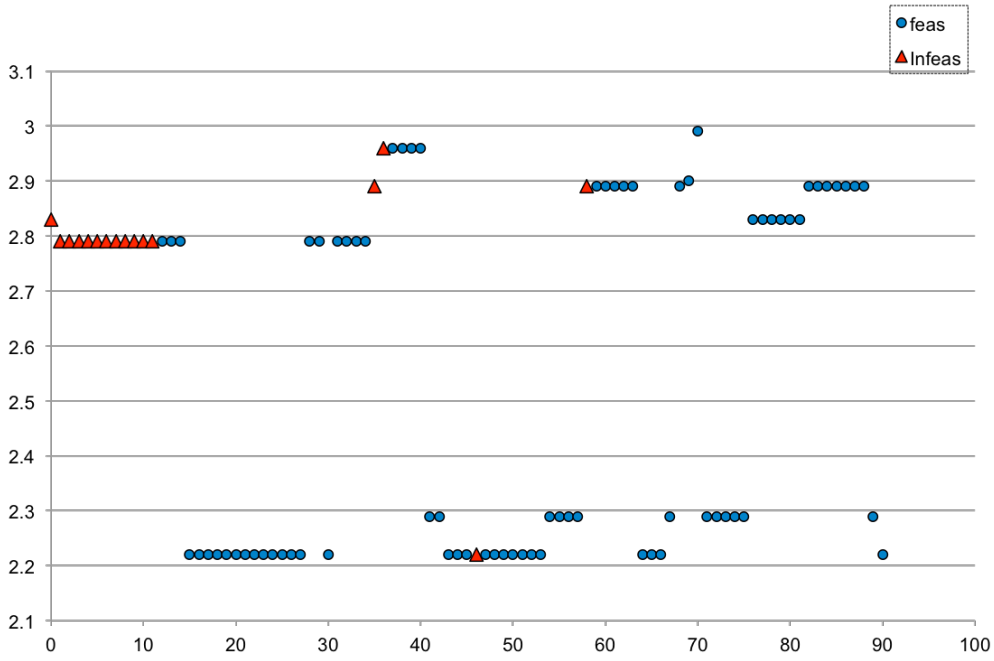

Figure 10: Behavior of TS in a single 300 × 4 instance.

6 Conclusions

In this paper we have presented an improved tabu search algorithm for a maximum dispersion territory design problem arising in the recollection of waste electric and electronic equipment. The proposed metaheuristic is enhanced with several algorithmic features. All components and strategies were empirically evaluated obtaining excellent results. Particularly, the strategic oscillation component proved extremely useful for further improving the quality of the solutions. The results indicated the overall efficiency of the algorithm, including significant improvements over the best solutions reported by a previously presented heuristic based on GRASP.

Acknowledgments: The first author was supported by the Mexican National Council for Science and Technology (CONACYT grants CB05-1-48499Y and CB11-1-166397) and by UANL through its Scientific and Technological Research Support Program (grants UANL-PAICYT CE012-09, IT511-10, CE728-11, and CE331-15.) The research of the second author was supported by the Tecnológico de Monterrey Research Group in Industrial Engineering and Numerical Methods 0822B01006. The third author was supported by a scholarship for graduate studies from CONACyT and UANL.

References

[1] B. Bozkaya, E. Erkut, and G. Laporte. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research*, 144(1):12–26, 2003.

[2] J. C. Duque, R. Ramos, and J. Suriñach. Supervised regionalization methods: A survey. *International Regional Science Review*, 30(3):195–220, 2007.

[3] E. Fernández, J. Kalcsics, and S. Nickel. The maximum dispersion problem. *Omega*, 41(4):721–730, 2013.

[4] E. Fernández, J. Kalcsics, S. Nickel, and R. Z. Ríos-Mercado. A novel maximum dispersion territory design model arising in the implementation of the WEEE-directive. *Journal of the Operational Research Society*, 61(3):503–514, 2010.

[5] P. Georgiadis and M. Besiou. Environmental and economical sustainability of WEEE closed-loop supply chains with recycling: A system dynamics analysis. *International Journal of Advanced Manufacturing Technology*, 47(5–8):475–493, 2010.

[6] F. Glover and J.-K. Hao. The case for strategic oscillation. *Annals of Operations Research*, 183(1):163–173, 2011.

[7] F. Glover and M. Laguna. *Tabu Search*. Kluwer, Boston, 1997.

[8] D. Hammond and P. Beullens. Closed-loop supply chain network equilibrium under legislation. *European Journal of Operational Research*, 183(2):895–908, 2007.

[9] W. He, G. Li, X. Ma, H. Wang, J. Huang, M. Xu, and C. Huang. WEEE recovery strategies and the WEEE treatment status in China. *Journal of Hazardous Materials*, 136(3):502–512, 2006.

[10] R. Hischier, P. Wäger, and J. Gauglhofer. 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, 2005.

[11] J. Kalcsics. Districting problems. In G. Laporte, S. Nickel, and F. Saldanha da Gama, editors, *Location Science*, chapter 23, pages 595–622. Springer, Cham, Switzerland, 2015. ISBN: 978-3-319-13110-8.

[12] S. C. Lee and L. H. Shih. A novel heuristic approach to determine compromise management for end-of-life electronic products. *Journal of the Operational Research Society*, 63(5):606–619, 2012.

[13] J. Mar-Ortiz, B. Adenso-Diaz, and J. L. González-Velarde. Design of a recovery network for WEEE collection: The case of Galicia, Spain. *Journal of the Operational Research Society*, 62(8):1471–1484, 2011.

[14] J. Mar-Ortiz, J. L. González-Velarde, and B. J. Adenso-Díaz. Designing routes for WEEE collection: The vehicle routing problem with split loads and date windows. *Journal of Heuristics*, 19(2):103–127, 2013.

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

[16] F. Ricca, A. Scozzari, and B. Simeone. Political districting: From classical models to recent approaches. *Annals of Operations Research*, 204(1):271–299, 2013.

[17] C. Rudăreanu. Waste electrical and electronic equipment (WEEE) management in Europe. *Economics, Management, and Financial Markets*, 8(3):119–125, 2013.

[18] C. Rudăreanu. New challenges for the WEEE management system in Romania as a result of the recast of the WEEE directive. *Contemporary Readings in Law and Social Justice*, 6(1):119–125, 2014.

[19] W.-H. Tsai and S.-J. Hung. Treatment and recycling system optimisation with activity-based costing in WEEE reverse logistics management: An environmental supply chain perspective. *International Journal of Production Research*, 47(19):5391–5420, 2009.

[20] G. Walther and T. Spengler. Impact of WEEE-directive on reverse logistics in Germany. *International Journal of Physical Distribution and Logistics Management*, 35(5):337–361, 2005.