DEPARTMENT OF OPERATIONS RESEARCH
UNIVERSITY OF AARHUS

Publication no. 2000/1

The Bicriterion Semi-obnoxious Location Problem (BSLP) Solved by an ϵ -Approximation

Anders J.V. Skriver
Kim Allan Andersen

ISSN 1398-8964

Department of Mathematical Sciences
Telephone: +45 8942 3188
E-mail: institut@imf.au.dk

Building 530, Ny Munkegade
DK-8000 Aarhus C, Denmark
URL: www.imf.au.dk

The Bicriterion Semi-obnoxious Location Problem (BSLP) Solved by an ϵ-Approximation

ANDERS J. V. SKRIVER [*] AND KIM ALLAN ANDERSEN
Department of Operations Research
University of Aarhus, building 530
Ny Munkegade
DK - 8000 Århus C
Denmark

February 7, 2000

Abstract

Locating an obnoxious (undesirable) facility is often modeled by the maximin or maxisum problem. But the obnoxious facility is often placed unrealisticly far away from the demand points (nodes), resulting in prohibitively high transportation cost/time. One solution is to model the problem as a semi-obnoxious location problem.

Here we model the problem as a bicriterion problem, not in advance determining the importance of the obnoxious objective compared to the cost/time objective.

We consider this model for both the planar and the network case. The two problems are solved by an approximation algorithm, and the models are briefly compared on a real-life example.

Keywords: Multiple criteria analysis, Semi-obnoxious, Location, Planar, Networks.

1 Introduction

In the two traditional Single Facility Location Problems (SFLP), a new facility is located (placed) so as to minimize transportation costs (minisum), or as to minimize the distance to the farthest customer (minimax). In the minisum problem we sum all the distances between the new facility and the customers, multiplied by a weight depending on the individual customer. In the minimax problem we minimize the largest weighted distance. The minisum model can be relevant when locating a warehouse and the minimax model can be used to locate a fire station. These models are presented in Love *et al.* [14] and Francis *et al.* [8], both including many references. The obnoxious location problem is a more recent class of problems, where the two most common are the maxisum and maximin

[*]Corresponding author. Email: ajs@imf.au.dk

models. When locating an obnoxious (undesirable) facility the goal is to place it as far from the existing facilities (demand points, customers) as possible. See Erkut and Neuman [6] for a review.

There is little literature combining the desirable and the obnoxious facility location models. In this paper we model the combined problem as a Bicriterion Semi-obnoxious Location Problem (BSLP). One objective function is obnoxious and one is desirable. We also consider both the network case and the planar case of the problem. In biobjective optimization our goal is to find the set of efficient solutions. These solutions are such that there does not exist another solution that has a better value in one objective without having a worse value in the other objective. The concept of efficient solutions is the same as Pareto optimal solutions. In the network case, where the demand points are nodes in a network and we try to locate the new facility in a node or on an edge, we have found no references, but ongoing research is presented in Hamacher *et al.* [11]. In the planar case, where the feasible locations are in R^2, we have found only three references, namely two papers by Brimberg and Juel, [1] and [2], and a paper by Carrizosa *et al.* [4].

In the bicriterion model, developed in the first paper by Brimberg and Juel [1], the first objective is the minisum objective and the second objective (the obnoxious criterion) is the minisum objective, where the Euclidean distance is raised to a negative power. It is proposed to solve the problem (finding the efficient solutions) in two steps. First a convex combination with parameter $\lambda \in [0, 1]$ of the two objectives (weighting method, Steuer [16]) is formed. The resulting objective is neither convex nor concave. By varying λ a trajectory of efficient solutions may be determined. In the paper an algorithm based on this is outlined. A numerical example is presented.

In the second paper by Brimberg and Juel [2] a different bicriterion model is considered. In this model the first objective is again the minisum objective, but the second objective (obnoxious) is now the maximin objective. They present two different solution methods for this model, but only one of them is guaranteed to find the complete set of efficient solutions.

In the bicriterion model developed in the third paper by Carrizosa *et al.* [4], the first objective (the obnoxious criterion) is modeled as the maxisum, and the second objective is modeled as the minisum problem. A solution procedure based on the BSSS (Big Square Small Square) approach is suggested. The procedure finds an approximation of the set of efficient solutions but no computational experience is reported.

The theory of the planar and network models is quite different, and the two models are not often compared, even though they try to describe the same real-life problem. We apply the two models on a real-life example in Section 4.

Next we present the basic model for the Bicriterion Semi-obnoxious Location Problem (BSLP). We assume that there are n existing facilities (demand points). In the planar case they are denoted $a_j = (a_{j1}, a_{j2}),\ j = 1, \dots, n$. In the network case they are denoted $v_1, v_2, \dots, v_n$. We want to place a new facility at location x in order to minimize both the (transportation) costs and the obnoxiousness. Let S denote the set of feasible solutions, $f(x)$ the obnoxious objective function and $g(x)$ the cost objective function. The general model looks as follows:

$$\begin{array}{ll} \min & f(x) \\ \min & g(x) \\ \text{s.t.} & \\ & x \in S \end{array} \tag{1}$$

We assume f depends negatively on the distance function and g depends positively on the distance function. This means, when we increase the distance between the new facility and an existing facility, this will have a decreasing effect on f and an increasing effect on g, e.g. less obnoxiousness but higher transportation costs.

Since it is very unlikely that there exists a location x which is an optimal solution for both objectives, we have to settle with something less, namely efficient (Pareto Optimal) solutions. For a reference in multicriteria analysis see Steuer [16].

Definition 1 *A feasible solution x to (1) is **efficient** iff there does not exist another feasible solution $\bar{x}$ to (1) such that $f(\bar{x}) \leq f(x),\ g(\bar{x}) \leq g(x)$ and $(f(\bar{x}), g(\bar{x})) \neq (f(x), g(x))$. Otherwise x is **inefficient**.*

The set of efficient solutions are denoted $\mathcal{X}_{Par}$. Efficiency is defined in the decision space. There is a natural counterpart in the criterion space. The feasible region in criterion space is denoted by $\mathcal{Z}$ and is given by $\mathcal{Z} = \{z(x) \in R^2 | z(x) = (f(x), g(x)),\ x \text{ is feasible in } (1)\}$.

Definition 2 *$z(x) \in \mathcal{Z}$ is a **nondominated** criterion vector iff x is an efficient solution to (1). Otherwise $z(x)$ is a **dominated** criterion vector.*

We note that several efficient solutions may correspond to the same nondominated criterion vector. The set of nondominated criterion vectors is denoted $\mathcal{Z}_{Par}$, where $\mathcal{Z}_{Par} = z(\mathcal{X}_{Par})$.

As mentioned we consider two cases of the problem. The planar case, denoted BSPLP, where the feasible solutions form a region in the plane, and the network case, denoted BSNLP, where the set of demand points are vertices in a network. For the BSNLP we

Position	Meaning	Usage (Examples)
1	number of new facilities	
2	type of problem	**P** planar location problem **D** discrete location problem **G** network location problem
3	special assumptions and restrictions	$w_m = 1$ all weights are equal $\mathcal{R}$ a forbidden region
4	type of distance function	l_1 Manhattan metric $(l^p)^{-b}$ l^p norm to negative power $d(\mathcal{V},\mathcal{V})$ node to node distance $d(\mathcal{V},G)$ node to point distance
5	type of objective function	$\sum$ Median problem $\sum_{obnox}$ Anti-Median problem $\max$ Center problem $\max_{obnox}$ Anti-Center problem

Table 1: Classification scheme for location problems.

allow solutions to be both the nodes and the points on the edges. This is often referred to as absolute location (on networks).

The BSPLP is solved using the Big Square Small Square (BSSS) method described by Hansen *et al.* [12], and the idea of this method has been applied to solve the BSNLP as well. The method is described in Section 2.1 for the planar case and in Section 3.1 for the network case.

Below we introduce a classification scheme for location problems, which should help to get an overview over the manifold area of location problems. We use a scheme which is analogous to the one introduced successfully in scheduling theory. The presented scheme for location problems was developed in Hamacher and Nickel [10] and Hamacher *et al.* [9]. We have the following five position classification

$$pos1/pos2/pos3/pos4/pos5\ ,$$

where the meaning of each position is explained in Table 1.

If we do not make any special assumptions in a position, we indicate this by a $\bullet$.

The remaining part of the paper is organized as follows. In Section 2 we describe the BSPLP and the solution approximation algorithm, and in Section 3 the BSNLP problem and its solution method is described. In Section 4 an application of the two models is presented. Section 5 contains the conclusions.

2 The planar case : BSPLP

The Bicriterion Semi-obnoxious Planar Location Problem (BSPLP) is formulated in the following way. There are n facilities (demand points) located at points $a_1, a_2, \dots a_n$, and the objective is to locate a semi-obnoxious facility at x so as to minimize a weighted sum of the distances raised to a negative power, and to minimize the weighted sum of the distances between the existing facilities and the new facility. The first criterion may be thought of as a pollution effect and the second criterion as transportation costs.

$$\begin{array}{ll} \min & f(x) = \sum_j \, w_j^1 (\| x - a_j \|_p)^{-b}, \qquad b > 0 \\ \min & g(x) = \sum_j \, w_j^2 \, \| x - a_j \|_p \\ \text{s.t.} & \\ & x \in S \end{array} \tag{2}$$

where $\| x - a_j \|_p = (|x_1 - a_{j1}|^p + |x_2 - a_{j2}|^p)^{1/p}$ is the usual l^p norm, $p \geq 1$.

This problem is classified as $1/P/\bullet/((l^p)^{-b}, l^p)/(\sum_{obnox}, \sum)_{Par}$ using the classification scheme from Table 1.

We prefer this obnoxious function, because it minimizes the overall obnoxiousness when far from a demand-point, but reflects the local effects when close to a demand-point. Corresponding to this objective we use the weights w^1. The second objective is the standard formulation for locating an attractive facility by minimizing the weighted sum of the distances (called minisum or median). Please note that we use weights w^2 with this objective, so that the two objectives may be weighted differently with respect to each of the n demand points. We assume that all weights are nonnegative.

If we are modeling where to place a new airport (the example in Section 4), the first weight w_j^1 may depend on the population at demand point j (e.g. city), and the second weight w_j^2 may be the expected number of passengers on a yearly basis from demand point j. S is the set of feasible solutions, often assumed to be a compact set. For references on location theory see Love *et al.* [14].

An obvious question for this model would be, if all feasible points are efficient? The answer is that there does indeed exist examples where all feasible points are efficient, but that will probably not be the case in a realistic set-up.

If the feasible region is a compact set, and we only have a single demand point with strictly positive weights for both objectives, the whole feasible region, except the demand point, will be efficient. The demand point will optimize the minisum objective with value zero, and the obnoxious objective will decrease as we move away from the demand point. Note that the obnoxious criterion is not defined at the demand points. As indicated by this simple example, situations may occur where the efficient region is not a curve. This may be a serious problem for the trajectory method used in Brimberg and Juel [1].

2.1 The idea of the Big Square Small Square (BSSS) algorithm

In this paper the idea behind the BSSS method will be applied to the BSPLP (and also to the BSNLP). Therefore we briefly review the method below.

Suppose that the feasible region S is contained in a disjoint union of squares of equal size. We put these squares into a list named ES. Next each of these squares are considered separately. Consider one of the squares, say Q_i. We divide Q_i into four sub-squares Q_{i1}, Q_{i12}, Q_{i3} and Q_{i4} of equal size. For each of these sub-squares, say Q_{i1}, lower bounds on the objective function values $(f(x), g(x)),\ x \in Q_{i1}$, are found. By comparing this lower bound with a sample set of objective function values (stored in a list called EFV) it may be determined that square Q_{i1} contains only inefficient points (this is done by the Dominance Check Routine DCR(Q_{i1})). If this is the case square Q_{i1} is called an inefficient square and may be deleted from further consideration. The squares that cannot be classified as inefficient are put into the ES list and will later be divided further into four new sub-squares. The process continues until the side-lengths of all the remaining squares (those that are not classified as inefficient) in ES are below some pre-specified value ϵ. The idea is illustrated in Figure 1.

A few comments on the procedure are appropriate. The sample list of objective function values kept in (the sorted) list EFV (Efficient Function Value) are used to dominate sub-squares with poor objective function value bounds. Therefore the values should in a way represent the objectives' behaviour over the feasible region. This is done by calculating objective function values in the centers of all the squares, and then deleting pairs of objective function values being dominated by other objective function values in the EFV list. It is also essential that we use good lower bounds for the objective function values over the squares. If the bounds are poor, the convergence of the algorithm may be slow, because we will end up with a large number of squares. Fortunately, we have found good bounds that work under mild conditions. These bounds are explained in detail in Sections 2.3 and 2.4. Finally, we need to check if a square is contained in the feasible region, is overlapping the region or is outside the region. For a discussion of this issue we refer to the paper by Hansen *et al.* [12] and Section 2.2.

The output from the algorithm is an ordered set of "efficient" squares. These squares can be associated with a certain objective function value, to illustrate the trade-off between the two objectives. This can be done by giving the squares a color corresponding to the

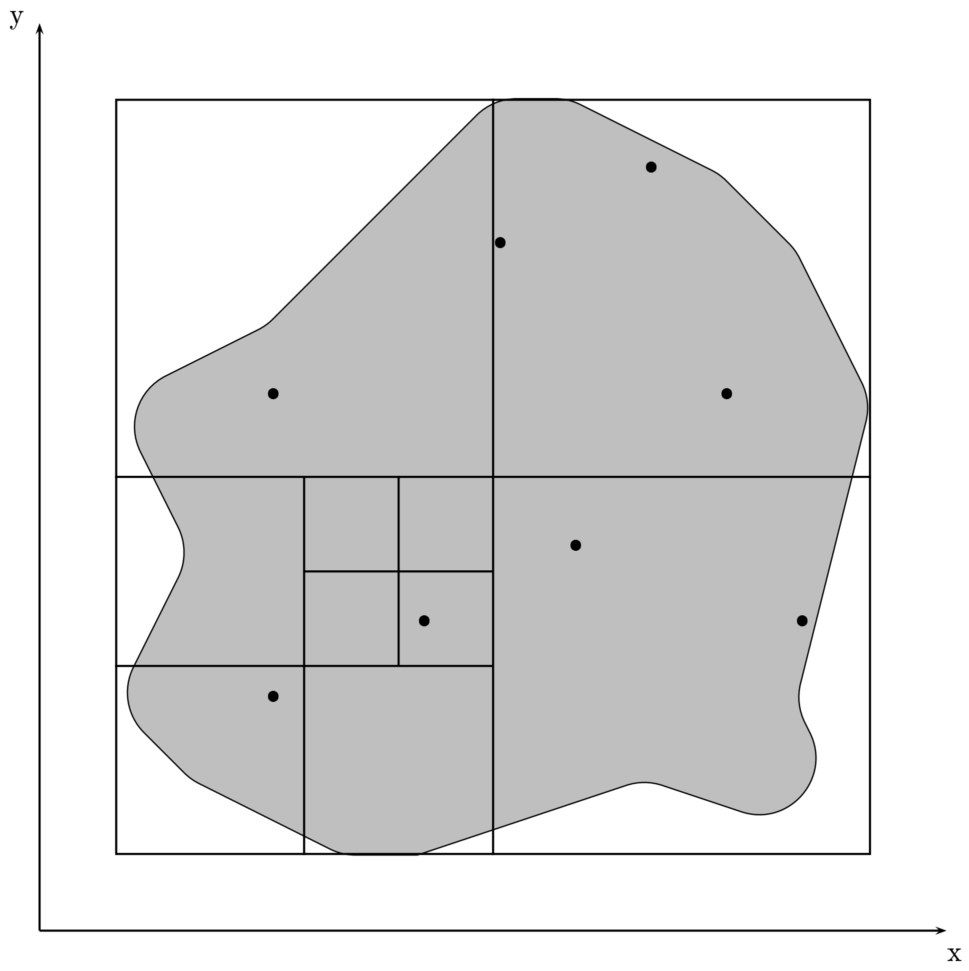

Figure 1: BSSS idea

value of the first objective. This will illustrate how one objective improves as the other gets worse, and visualize the objective function values being favoured in the different efficient regions.

2.2 Square approximation of the feasible region S

In this section we will use a slightly different approach than the one suggested by Hansen *et al.* [12] and described previously in section 2.1.

We assume that the feasible set S is a bounded set (usually it is assumed compact). We start with an approximation of S consisting of equal size squares. This way we can skip the feasibility tests, and for the accuracy of a realistic model it may not be a real loss (but of course, we could approximate S more closely by adding linear constraints).

2.3 Calculating lower bounds

In order to calculate lower bounds on the two objectives, we use an approximation of the weighted distances. This distance approximation is illustrated in Figure 2 for the l^2 norm. The lower bound for the distance is found in Hansen *et al.* [12], and the upper bound for the distance is an obvious extension of the same idea, found in Hansen *et al.* [13].

The plane is divided into 9 regions, obtained by extending the four sides of Q_i. The regions are the square Q_i, the four side regions, and the four corner regions. The square Q_i will be in the center.

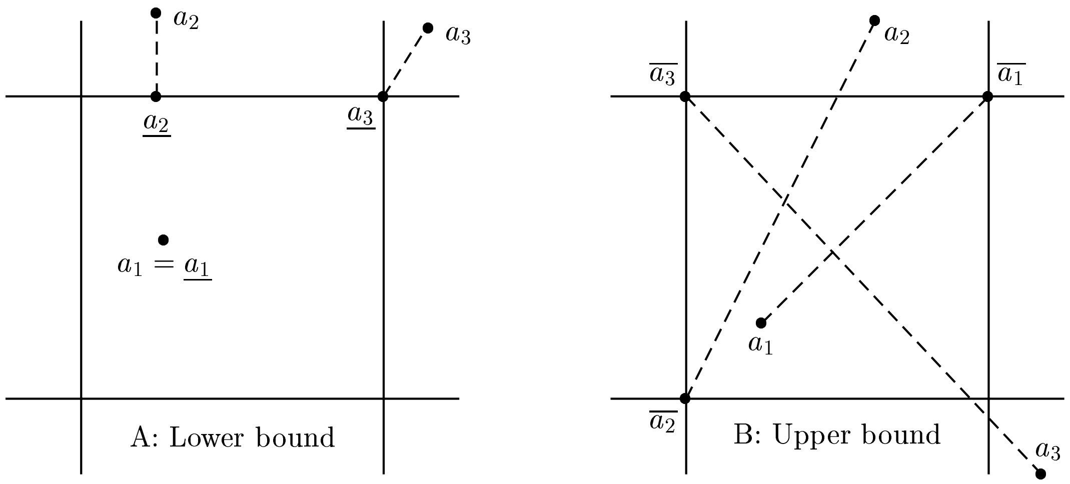

Figure 2: Lower and upper bounds on the distances.

Now let a_j be a particular location. With this location we associate a closest point $\underline{a_j} \in Q_i$ and a furthest point $\overline{a_j} \in Q_i$, see Figure 2. We may then calculate a lower bound on the values of f and g in Q_i as follows:

$$\begin{array}{rcll} \underline{f}(Q_i) & = & \sum_j w_j^1(\| \overline{a_j} - a_j \|_p)^{-b} & \text{Case B in Figure 2} \\ \underline{g}(Q_i) & = & \sum_j w_j^2 \| \underline{a_j} - a_j \|_p & \text{Case A in Figure 2} \end{array}$$

Clearly, $(\underline{f}(Q_i), \underline{g}(Q_i) \leq (\min_{x \in Q_i} f(x), \min_{y \in Q_i} g(y))$. Therefore we can use the bound $\underline{z}(Q_i) = (\underline{f}(Q_i), \underline{g}(Q_i))$ for efficiency checking in the algorithm. If we at some point have found a sample value $x \in S$, such that $(f(x), g(x)) \leq (\underline{f}(Q_i), \underline{g}(Q_i))$, then, clearly all points in Q_i are dominated by x. It follows that square Q_i contains only inefficient points. Therefore it is not necessary to consider Q_i anymore. This bound approach can be used for any $p \in [1; \infty]$. Please note that the bounds obviously converge when the squares get smaller.

2.4 Exact lower bound

Since the minisum objective is a nice convex function, it is possible to calculate an exact lower bound for the squares in most situations. The level sets of a convex function are convex sets, and the gradient can therefore be used as follows.

For a square Q_i with corners c_1, c_2, c_3 and c_4, find the corner c_h with the minimum function value $g(c_h)$. If the direction of steepest descent "**points away**" from the square Q_i, then the lower bound $\underline{g}(Q_i)$ is exactly $g(c_h)$. By "pointing away" we mean that the direction of steepest descent has an angle of at least 90 degrees with the sides of Q_i, see case A in Figure 3. If this angle is less than 90 degrees, the minimum value over Q_i is not in c_h, but on the line segment between c_h and the corner, the direction points out, see case B in Figure 3. Finally, if the direction points **into** Q_i, the minimum value is **not** in c_h but inside Q_i.

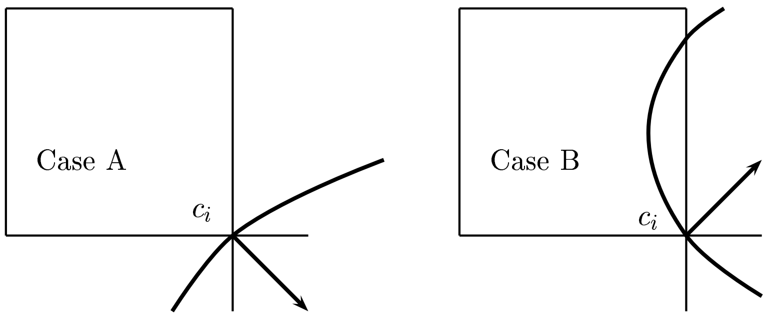

Figure 3: Exact lower bound, depending on directional derivative

From the above, an exact lower bound can easily be computed, if the directional derivative points away from the square. We only need to compute four function values and the directional derivative in the minimum value corner. Case A in figure 3 will occur in most evaluations, but not in all.

The directional derivative $g'(x_0, y)$ of g at $x_0 \in S$ in the direction y is defined as follows:

$$g'(x_0, y) = \nabla\, g(x_0) \cdot y$$

where $\nabla g(x_0)$ is the gradient of g evaluated in x_0.

If we consider the l^2 norm, the gradient looks as follows:

$$\nabla\, g(x_0) = \left(\sum_j \frac{w_j^2 (x_{01} - a_{j1})}{\parallel x_0 - a_j \parallel}, \sum_j \frac{w_j^2 (x_{02} - a_{j2})}{\parallel x_0 - a_j \parallel} \right)$$

This reveals the well-known problem; if x_0 is at a demand point, the gradient is undefined because of the numerator being zero. In this case we also have to use the lower bound of Section 2.3. Alternatively we could apply the hyperbolic approximation (see Love *et al.* [14]).

A computation of the gradient of the general l^p norm, for $p \in]1;\infty[$ is given below along with the gradient for the l^1 and l^∞ norm. All these expressions are well-defined, and can be easily computed. $sign(x)$ is the sign function, which is one if x is positive, and minus one if x is negative. We therefore conclude that an exact lower bound exists for the l^p norm $\forall p \in [1;\infty]$.

$$\begin{aligned} \nabla\, g(x_0) &= \left(\sum_j \frac{sign(x_{01}-a_{j1}) w_j^2 |x_{01}-a_{j1}|^{p-1}}{(|x_{01}-a_{j1}|^p + |x_{02}-a_{j2}|^p)^{\frac{p-1}{p}}}, \right. \\ & \left. \sum_j \frac{sign(x_{02}-a_{j2}) w_j^2 |x_{02}-a_{j2}|^{p-1}}{(|x_{01}-a_{j1}|^p + |x_{02}-a_{j2}|^p)^{\frac{p-1}{p}}} \right) \quad 1 < p < \infty \\ \nabla\, g(x_0) &= \left(\sum_j sign(x_{01}-a_{j1}) w_j^2, \sum_j sign(x_{02}-a_{j2}) w_j^2 \right) \quad p = 1 \end{aligned}$$

Let us remind the reader of the definition of the l^∞ norm.

$$l^\infty(x_0, a_j) = \max\{|x_{01}-a_{j1}|, |x_{02}-a_{j2}|\}$$

This maximum is attained in either the horizontal or the vertical direction, so we may refer to the maximum being taken in the first or the second coordinate. The partial derivative of the l^∞ norm is the following:

$$\frac{\partial l^\infty(x_0, a_j)}{\partial x_{0i}} = \begin{cases} 0 & \text{Maximum } \textbf{not} \text{ in coordinate } i \\ sign(x_{0i}-a_{ji}) & \text{Maximum in coordinate } i \end{cases}$$

We now see that the partial derivative depends on whether the maximum is taken in the first or the second coordinate. To get a compact form of the gradient, lets define two index sets as follows. N is the set of the indexes of the n demand points, that is $N = \{1, 2, \ldots n\}$. Now define A as the set of indices where the maximum is in the first coordinate. We can now write the gradient for the l^∞ norm.

$$\nabla\, g(x_0) = \left(\sum_{j \in A} sign(x_{01}-a_{j1}) w_j^2, \sum_{j \in N \setminus A} sign(x_{02}-a_{j2}) w_j^2 \right)$$

This gradient is easily computed by simple norm evaluations.

The only assumption needed for the exact lower bound to be valid, is that the level sets are convex.

2.5 BSSS algorithm for the BSPLP

Notation:

Q_i	Square number i
$\underline{z}(Q_i) = (\underline{f}(Q_i), \underline{g}(Q_i))$	Lower bounds for Q_i.
ES	List of Efficient Squares. Note that this is only a name for squares that have not been proven inefficient.
ECL	Efficient Candidate List (of squares of equal size). It consists of the four sub squares of all the squares in ES.
EFV	List of Efficient Function Values. Function values are calculated at different points in the feasible region, and the nondominated ones (at this time in the routine) are in this list.
DCR(Q_i)	Dominance Check Routine for Q_i (with EFV). Is briefly explained in Section 2.1.

The idea for the DCR routine was found in [3], and earlier used by the authors in [15].

Algorithm 2.5:

```
1. Initialize

      Find an equal size square approximation $Q_1, Q_2, \dots, Q_N$ of S
      Put $Q_i$ in ES $\forall i = 1, 2, \dots, N$.
      Let $L$ be the length of a side of $Q_1$
      Define the tolerance level $\epsilon$

2. Creating New Squares

      For each $Q_i \in$ ES do
         Create 4 sub-squares $Q_j, j = 1, 2, 3, 4$, put the $Q_j$'s in ECL and delete $Q_i$
         from ES
      Set $L = \frac{L}{2}$

3. Efficiency Update

      Update EFV by calculating some function values from the $Q_j$'s
      For each $Q_j \in$ ECL do
         Calculate $\underline{z}(Q_j) = (\underline{f}(Q_j), \underline{g}(Q_j))$ using exact lower bounds when possible
         Make DCR($Q_j$) with EFV
         If $Q_j$ is efficient compared with EFV then add $Q_j$ to ES

4. Termination Test
```

If $L < \epsilon$ Terminate with ES as the solution list

Else go to Step 2

3 The network case : BSNLP

In this section we adapt the BSSS method to the network case. However, instead of dividing big squares into smaller, we divide edges into sub-edges. This will be explained in details in Section 3.1. Assume we have an undirected connected network $G(\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V} = \{v_1, v_2, \dots, v_n\}$ where $|\mathcal{V}| = n$ nodes, and a finite set of edges (arcs) $\mathcal{E} = \{(v_i, v_j), (v_k, v_l), \dots, (v_p, v_q)\}$ with $|\mathcal{E}| = m$. Edges may also be denoted by e. The length of edge (v_i, v_j) is denoted by l_{ij}. Each node v_j carries two weights (w_j^1, w_j^2), one for the obnoxious criterion and one for the desirable criterion. The problem is to find the set of efficient points x on G called $\mathcal{X}_{Par}$. A point x can be located both at a node or on an edge.

The model is a combination of two well-known single criterion models, namely the maxisum and the minisum models. The two objectives are obviously negatively correlated and are often referred to as the weighted anti-median and the weighted median of a network. The Bicriterion Semi-obnoxious Network Location Problem, BSNLP, is then:

$$\begin{array}{ll} \max & f(x) = \sum_j \, w_j^1 \, d(x, v_j) \\ \min & g(x) = \sum_j \, w_j^2 \, d(x, v_j) \\ \text{s.t.} & \\ & x \in G(\mathcal{V}, \mathcal{E}) \end{array} \tag{3}$$

where $d(x, v_j)$ is the shortest distance from point x to node v_j.

This problem is classified by $1/G/\bullet/d(\mathcal{V}, G)/(\sum_{obnox}, \sum)_{Par}$ using the scheme presented in Table 1. The solution procedure is described shortly in Section 3.1 and the algorithm is presented in Section 3.4. An exact algorithm for this simple model is presented in Hamacher, Nickel and Skriver [11]. The approximation algorithm, however, is a very general and intuitive approach and can be used for more complicated objective functions.

Let us consider the two objectives separately. In Church and Garfinkel [5] the maxisum has been developed. They introduce the concept of bottleneck points, and refer to nodes with degree one as dangling nodes (often called pendant nodes). A bottleneck-point B is a point that has two different shortest paths to a particular node v_j. They show that among the optimal solutions a point x exists that is either a bottleneck-point or a pendant node. Furthermore they describe (easy) procedures for obtaining the bottleneck-points.

This is true because the weighted-sum objective is a piecewise linear concave function on the edges, with break-points only in the bottleneck-points. Note that the optimum need not be unique, it may be a sub-edge between two (or more) bottleneck-points.

It is well-known that an optimal solution to the minisum problem is found in a node. To find an optimal node we first find the distance matrix $D = \{d_{ij}\}$ of shortest distances $d_{ij} = d(v_i, v_j)$ between all pairs of nodes v_i and v_j. This matrix can be calculated in $O(n^3)$ running time using Floyd's algorithm or by applying Dijkstra's algorithm to all n nodes. For details on these graph procedures see Thulasiraman and Swamy [17]. Then the rows of the distance matrix D are multiplied by the corresponding weights. The row with the smallest weighted sum corresponds to the minisum optimum node. For further details on the minisum problem see Evans and Minieka [7] or Francis, McGinnis and White [8].

3.1 The Edge Dividing algorithm

The idea of the Edge Dividing (ED) algorithm is similar to the idea behind the BSSS algorithm. First we divide each edge into two sub-edges. Then bounds on the objective function values on each sub-edge are calculated. Furthermore, a sample set of objective function values are calculated. If the bounds calculated for a sub-edge are dominated by one (or more) of the sample set objective function values then the sub-edge is dominated and may be deleted from further consideration.

The bounds are derived in detail in Sections 3.2 and 3.3. The sample set of objective function values are calculated in the middle (center) of the sub-edges. Nondominated criterion values are kept in the EFV list.

Please note that only an approximation of the efficient set $\mathcal{X}_{Par}$ is found.

The output from the algorithm is an ordered set of "efficient" sub-edges. This general procedure, however, has a few disadvantages. The efficient set (or part of it) may be an edge-segment. This sub-edge will obviously remain in the ES list, but the sub-edge will be divided into sub-edges again and again. This reveals that the ES set will probably almost double in size, when we half the ϵ value. This can in fact be used as an alternative stopping criterion.

3.2 Calculating upper and lower bounds

We need both upper and lower bounds on the distance $d(x, v_j)$, where x can be any point on the edge (or sub-edge) e_i. We refer to the lower bound of this distance by $\underline{d(e_i, v_j)}$ and to the upper bound by $\overline{d(e_i, v_j)}$.

The upper-bound may be calculated as follows:

$$\overline{d(e_i, v_j)} = \min\{d(v_j, v_h) + d(v_h, x_h), d(v_j, v_k) + d(v_k, x_k)\} + d(x_h, x_k)$$

and the lower-bound may be calculated as follows:

$$\underline{d(e_i, v_j)} = \min\{d(v_j, v_h) + d(v_h, x_h), d(v_j, v_k) + d(v_k, x_k)\}$$

These two bounds can be easily calculated as illustrated in Figure 4, whenever the distance matrix D is available. The bounds are similar to those found in Hansen *et al.* [13] for a square.

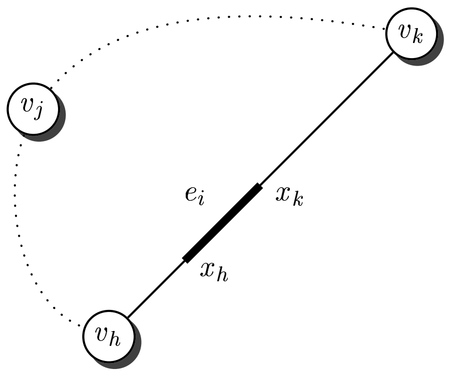

Figure 4: Calculating distance bounds.

Using these bounds we can calculate upper and lower bounds on the objective function values as

$$\begin{aligned} \overline{f}(e_i) &= \sum_j w_j^1 \, \overline{d(e_i, v_j)} \\ \underline{g}(e_i) &= \sum_j w_j^2 \, \underline{d(e_i, v_j)} \end{aligned}$$

3.3 Exact bounds

In this section we derive some exact bounds. First we need to define the derivative of the distance function $d(x, v_k)$ on the edge (v_i, v_j).

$$\frac{\partial^+}{\partial x_{(v_i, v_j)}} d(x, v_k) = \begin{cases} 1 & \text{if } d(x, v_k) \text{ is increasing from } v_i \text{ to } v_j \\ -1 & \text{if } d(x, v_k) \text{ is decreasing from } v_i \text{ to } v_j \end{cases}$$

With the notation $\frac{\partial^+}{\partial x_{(v_i, v_j)}} d(x, v_k)$ we simply mean that we look in the direction from v_i towards v_j, and we want to know if the distance increases or decreases. The "+" indicates right derivative, so even in a bottleneck-point this derivative is well-defined.

If we define $\mathcal{I}$ as the set of node-indexes where the distance function increases, then $\mathcal{V} \setminus \mathcal{I}$ is the set of node-indexes where the distance function decreases corresponding to the point $x_{(v_i,v_j)}$. We can now define the derivative of the two criterions:

$$\frac{\partial^+}{\partial x_{(v_i,v_j)}} f(x) = \sum_{k \in \mathcal{I}} w_k^1 - \sum_{k \in \mathcal{V} \setminus \mathcal{I}} w_k^1 \quad \text{and} \quad \frac{\partial^+}{\partial x_{(v_i,v_j)}} g(x) = \sum_{k \in \mathcal{I}} w_k^2 - \sum_{k \in \mathcal{V} \setminus \mathcal{I}} w_k^2$$

The sums $\sum_j \; w_j^1 \; d(x, v_j)$ and $\sum_j \; w_j^2 \; d(x, v_j)$ are concave functions on an edge. This is the reason why the maxisum optimum is located in a bottleneck-point and the minisum optimum is located in a node. If we are looking at the sub-edge from x_h to x_k as illustrated in Figure 4, and the derivatives at the endpoints have the same sign, then an exact upper bound is simply the largest endpoint value. That is, if

$$sign\left(\frac{\partial^+}{\partial x_{(v_h,v_k)}} f(x_h)\right) = sign\left(\frac{\partial^+}{\partial x_{(v_h,v_k)}} f(x_k)\right)$$

then

$$\overline{f}(e_i) = \max\{f(x_h), f(x_k)\} \tag{4}$$

For the minisum criterion it is even easier, since the minimum is always in one of the (sub-edge) endpoints. So we always have an exact lower bound as follows.

$$\underline{g}(e_i) = \min\{g(x_h), g(x_k)\} \tag{5}$$

3.4 ED algorithm for the BSNLP

Notation:

e_i	Sub-edge number i
$z(e_i) = (\overline{f}(e_i), \underline{g}(e_i))$	Upper and lower bounds for e_i.
ES	List of Efficient Sub-edge. Note that this is only a name for sub-edges that have not been proven inefficient.
ECL	Efficient Candidate List (of sub-edges). It consists of the two sub-edges of all the sub-edges in ES.
EFV	List of Efficient Function Values. Function values are calculated at different points on the network, and the nondominated ones (at this time in the routine) are in this list.
DCR(e_i)	Dominance Check Routine for e_i (with EFV).
L	Length of a longest edge in ES.

Algorithm 3.4:

1. Initialize

   Find the shortest path matrix D.

```
    Put all edges $e_1, e_2, \dots, e_m$ in ECL.
    Let $L$ be the length of a longest edge in ECL.
    Define the tolerance level $\epsilon$.
    Calculate criterion values in all nodes to make an initial EFV list.
2. Efficiency Update
    For each $e_i \in$ ECL do
        Calculate $z(e_i) = (\overline{f}(e_i), \underline{g}(e_i))$ using (5) and the exact bound (4) is possible
        Make DCR($e_i$) with EFV
        If $e_i$ is Efficient compared with EFV then add $e_i$ to ES
    Update $L$ (as a longest sub-edge in ES)
3. Termination Test
    If $L < \epsilon$ Terminate with ES as the solution list
4. Creating New Sub-edges
    For each $e_i \in$ ES do
        Split $e_i$ into two sub-edges $e_{i1}$ and $e_{i2}$ of equal length.
        Add $e_{i1}$ and $e_{i2}$ to ECL and delete $e_i$ from ES
    Update EFV by calculating criterion values on the middle of all sub-edges $e_j$ in ECL
    Go to Step 2
```

4 An airport example

To illustrate the usefulness of the two models we present an application. Currently, there is a debate in Denmark about where to locate a new international airport in the mainland Jutland to replace an existing one. The existing airport is located at a small city called Tirstrup approximately 45 km. to the North-East of Århus, the largest city in Jutland (with about 215.000 inhabitants). The existing airport is located in an area where not many people are living and where not many companies are based. Also, the infrastructure in this area is not too good. For example it takes about 1 hour for people to go from Århus to Tirstrup. Many companies (and people) think that this is too much time to spend on transportation to the airport.

It is believed that a new international airport located not too far away from Århus would be attractive to a lot of companies (and people). However, customers (companies/people) living nearby Århus is more likely to use the new airport than customers living far away from Århus. Therefore, for the planar case, we will only consider a region of potential locations with x-coordinates between 60 and 140, and y-coordinates between 100 and 180, see Figure 5. Furthermore, we have divided Jutland into three zones, namely a 100 % zone, a 50 % zone, and a 20 % zone, see Figure 5. The weighting zones should reflect the fact that customers far from the chosen region will use the new airport less frequently than customers close by or within this region. These three weighting zones will be used when defining the transportation objectives later in this section. We have chosen 42 cities to represent the customers in Jutland, ranging in population from 2574 (Hanstholm) to 215587 (Århus) inhabitants as demand points. Distance is measured in kilometers, and the ϵ-value used is 0.15 km (150 meters) for both the planar case and the network case. Origo is placed on the German island of Sylt.

Next, let us describe the parameters for the two objective functions. For the planar model we have used the Euclidean distance and a b-value of two. For the network model, the distance is always the shortest distance in the network. The edge lengths are road distances collected from an intercity distance table. All input data is available from the corresponding author.

For the obnoxious criterion we have used weights $w_j^1 =$"population in city j". This is a simple form of letting the larger cities count more than the smaller cities.

For the transportation cost objective we have used weights $w_j^2 =$"population in city j multiplied by the weight of the zone in which the city is located". This means that cities nearby Århus count much more than cities far away from Århus reflecting the fact that customers far away from Århus is likely to use the new airport less frequently than customers living nearby Århus.

Whether the city population is an appropriate measure of passengers is not an issue here. The data for the example is presented in Table 2. The three dummy-nodes in Table 2 are introduced only to make the road-network more realistic, and are located right to the West of Århus.

First we present the results of the planar model. The norm to the negative power function is illustrated in Figure 6, covering the region of $[60, 140] \times [100, 180]$. The peaks indicate the cities, where the function values goes to infinity. As can be seen from Figure 6 it may be hard to find an exact lower bound for this function.

The minisum global optimum is attained in $(110, 145)$ with a value of $3,27 \cdot 10^7$. The

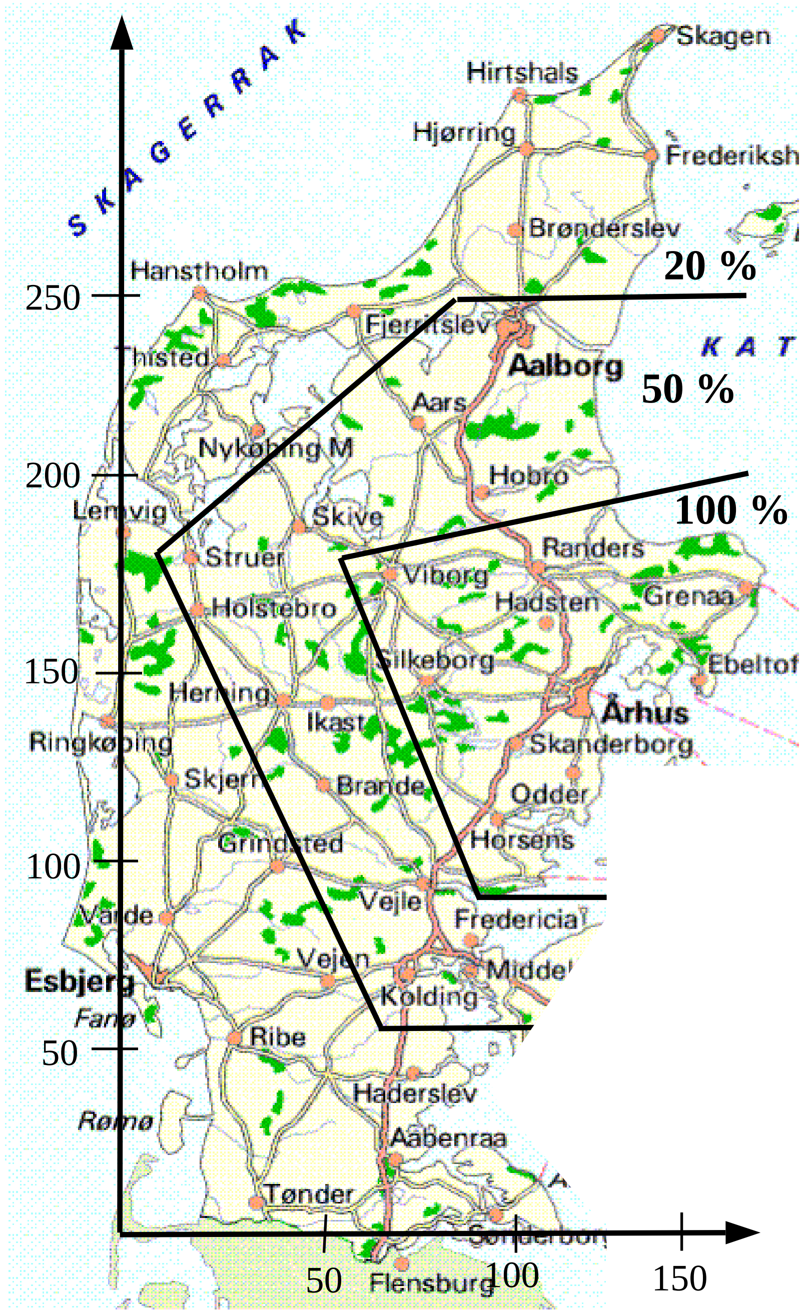

Figure 5: Jutland divided into three weighting-zones. Coordinates are in kilometers.

City (j)	a_{j1}	a_{j2}	w_j^1	w_j^2
Esbjerg	7.17	69.31	73422	14684.4
Tønder	34.416	8.126	8161	1632.2
Ribe	28.202	52.102	8046	1609.2
Kolding	72.178	68.832	53012	26506
Vejle	76.002	93.21	47839	23919.5
Horsens	95.122	110.418	48410	48410
Skanderborg	99.902	130.494	12067	12067
Århus	118.066	142.922	215587	215587
Randers	106.116	177.338	56123	56123
Viborg	67.876	175.904	31872	31872
Silkeborg	76.958	146.746	36762	36762
Ikast	52.102	141.01	14014	7007
Herning	40.63	141.966	29231	14615.5
Holstebro	18.642	166.344	30770	15385
Struer	17.208	181.206	11272	5636
Skive	44.454	188.332	20557	10278.5
Hadsten	108.028	162.042	6616	6616
Grenå	158.696	172.08	14441	14441
Hobro	91.298	196.936	10704	5352
Aars	74.568	216.056	7066	3533
Ålborg	98.468	240.434	119157	59578.5
Fredericia	88.43	78.392	29376	14688
Haderslev	74.09	42.064	21106	4221.2
Aabenrå	69.31	20.076	16218	3243.6
Vejen	52.58	66.442	8507	1701.4
Brønderslev	99.902	267.202	11365	2273
Hjørring	103.248	289.668	24889	4977.8
Frederikshavn	135.274	288.234	24768	4953.6
Bjerringbro	83.547	169.246	7201	7201
Varde	10.994	83.172	12478	2495.6
Grindsted	38.718	97.034	9497	1899.4
Skjern	11.95	119.978	6949	1389.8
Ringkøbing	-4.302	135.274	9166	1833.2
Brande	50.668	119.022	6214	3107
Lemvig	0	185.464	7302	1460.4
Nykøbing	33.46	212.71	9319	1863.8
Thisted	25.334	231.352	12609	2521.8
Hanstholm	19.12	249.516	2574	514.8
Fjerritslev	58.316	244.736	3332	666.4
Hirtshals	100.858	301.14	6949	1389.8
Skagen	136.708	315.958	10674	2134.8
Ebeltoft	146.746	144.834	4396	4396
Dummi North	113	152	0	0
Dummi West	109	144	0	0
Dummi South	109	137	0	0

Table 2: Locations $a_j = (a_{j1}, a_{j2})$ and weights (w_j^1, w_j^2) of 42 cities in Jutland.

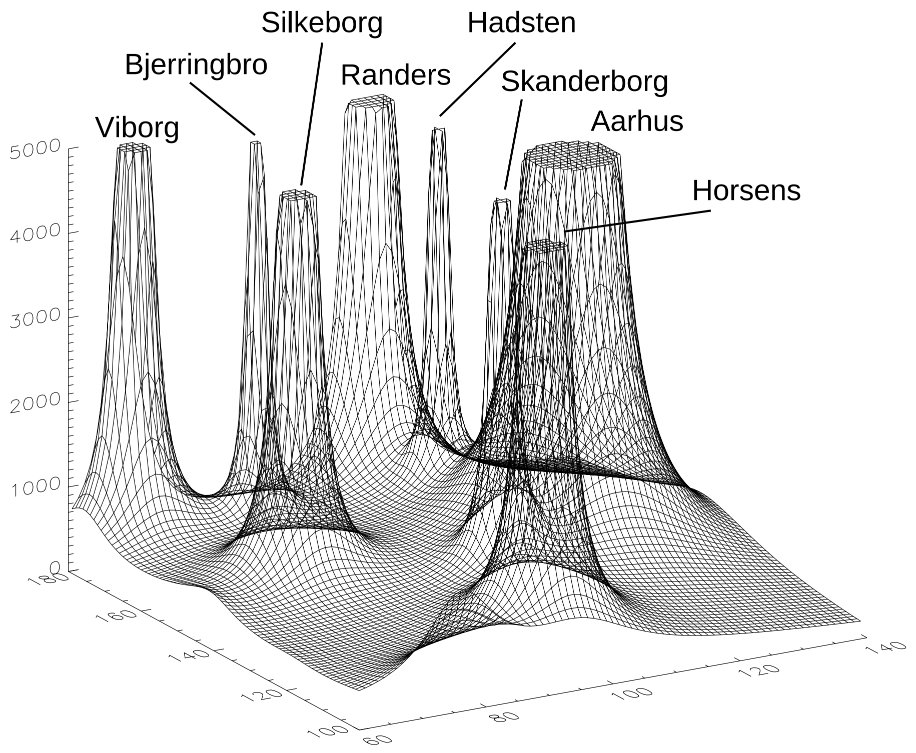

Figure 6: Surface-plot of the obnoxious objective-function.

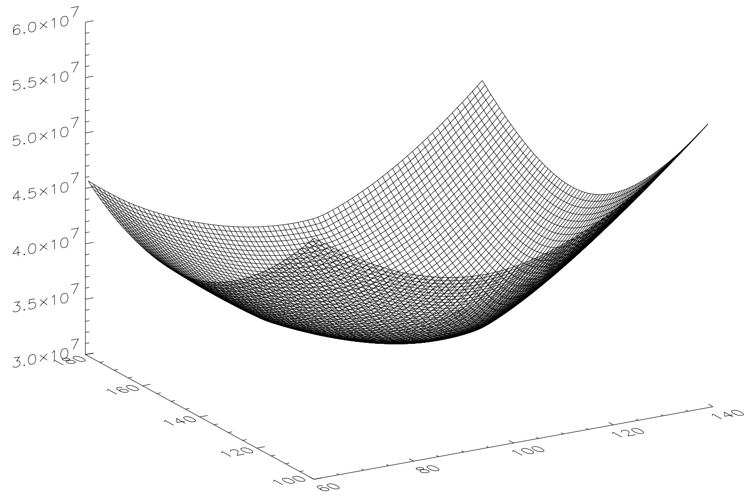

Figure 7: Surface-plot of the minisum objective-function

minisum function is plotted in Figure 7.

The efficient region is illustrated on the map in Figure 8. For clarity, we have drawn two minisum level curves. The inner level curve is minisum values 10 % above the global minisum minimum $(3,6 \cdot 10^7)$, and the outer level curve is 20 % above $(3,92 \cdot 10^7)$.

Figure 8 reveals three efficient regions. The central region just West-North-West of Århus containing the global minisum minimum, and with minisum-values within 10 % of the minimum. The central region reflects in which direction the obnoxious objective decreases, namely North-West. This region has the highest obnoxious values, ranging from 3400 (at the minisum optimum) to 675 in the North-Western part of the region. The South-West region has minisum values from approximately 10 % to 25 % above the minimum. This region has obnoxious values from 675 to 440. The last efficient region is the North-East region with minisum values more the 25 % above the minimum. This region has the lowest obnoxious values, below 440, simply because there are no major cities in this part of the country as can be seen in Figure 5. As a matter of fact the existing airport at Tirstrup is nearby this region.

Potential locations for a new airport should be found within these efficient regions.

Next we present the results of the network model. For this model only the resulting network, with the efficient sub-edges, are presented, see Figure 9. Notice that we do not restrict ourselves to a smaller region in this case. The minisum objective function has

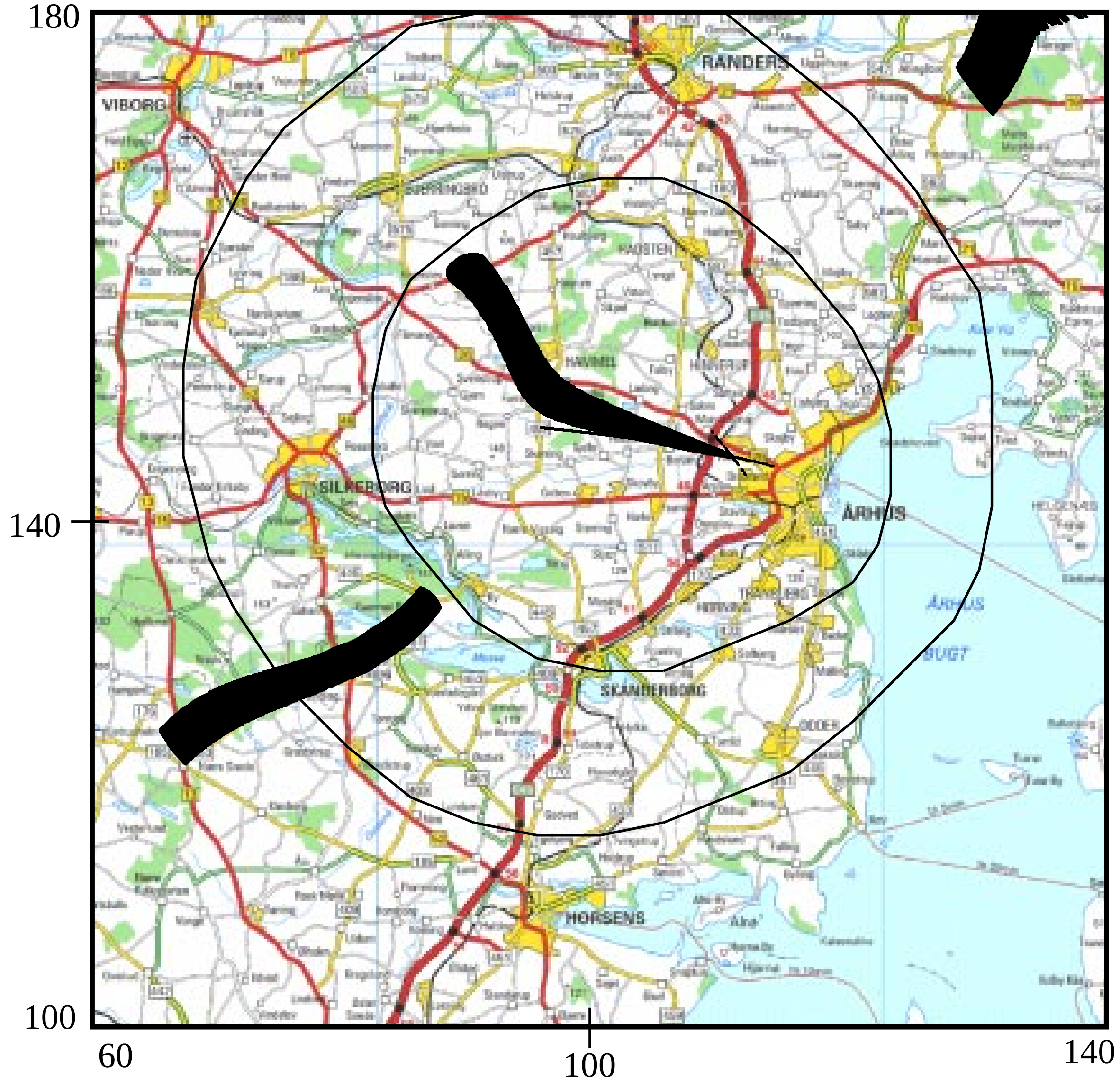

Figure 8: Efficient regions for airport location

its global minimum in the node representing the city of Århus. The maxisum objective function has its global maximum on the edge between Skagen and Hirtshals to the North of Jutland. Figure 9 reveals that the location of the new airport should be in an area North-East of Århus (in fact nearby Tirstrup) or in the North-Eastern part of Jutland.

It is interesting to learn that the planar and the network location models does not give the same answer to the problem of locating the new airport. If we just consider the region of interest in the planar model, one of the three efficient regions are in the same part of Jutland as the efficient region found in the network model. However, the planar location model proposes two efficient regions which are not proposed by the network model. Fortunately, the explanation to this is straightforward. The reason is simply that two distinct obnoxious criteria are used. In the planar case we minimize a weighted sum of the distances raised to a negative power whereas in the network model we maximize a weighted sum of distances. It is interesting to see the impact of the maxisum objective function. This is a global obnoxious function, which means that cities far from the new location count as much as cities close by. This effect is probably not reasonable, and therefore a maximin or norm to a negative power objective function may be more appropriate for the obnoxious objective in the network case as well.

5 Concluding remarks

In this paper we have set up two bicriterion location models for locating one obnoxious facility, namely one for the planar case and one for the network case. Efficient (well-working) solution algorithms based on the well-known BSSS algorithm has been proposed. Both models are easily extended to multiple criteria. All that needs to be changed is the merge operation, presently merging two dimensional sets.

Even though the planar and the network model may seem distinct in structure, they are designed to solve the same real-life problem. Often a combination of the two models would be preferable. For example, modeling air pollution such as noise makes most sense in the planar model, whereas the network model would be the correct description of a road network with distances or travel times as coefficients. One possible combination is to embed the network on top of the plane, so that each point on the network corresponds to a point in the plane, but not necessarily the other way around.

Another issue is the choice of obnoxious criterion functions. In the planar case we have used the negative power function also used in Brimberg and Juel [1], and for the network case we have used the maxisum function often used in the literature. They share the common feature that they depend negatively on the distance between the new location

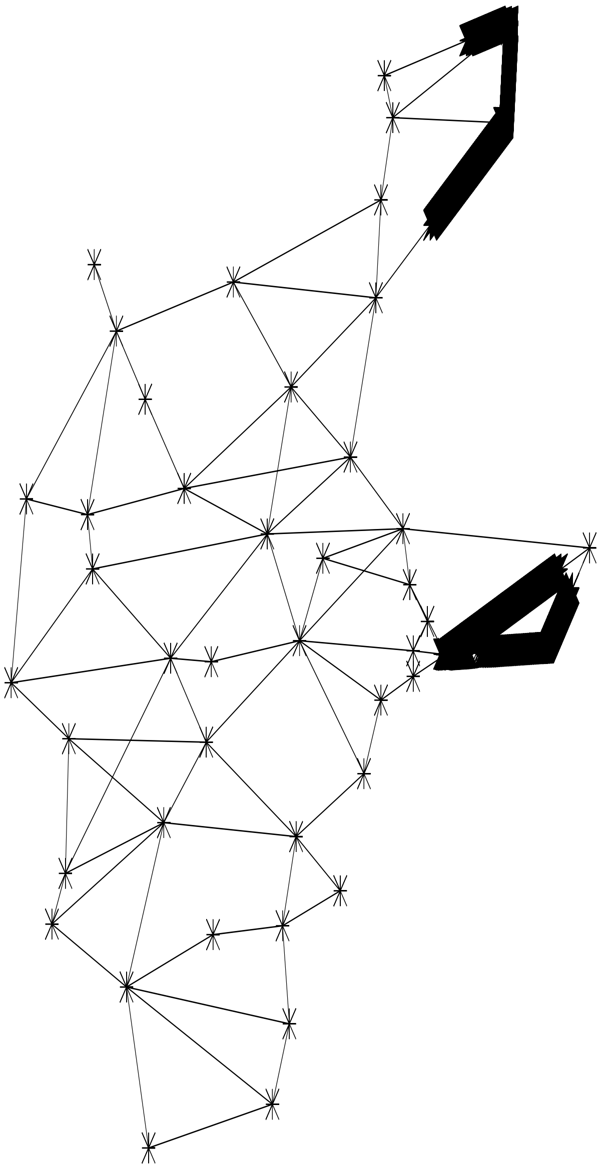

Figure 9: Road-network of Jutland. Bold parts constitute the efficient set.

and the existing location(s). Of course, many other functions may be used, and for more complicated functions, the approximation approach described in this paper may be the only applicable approach.

It may also be appropriate to have weights depending on distance. However, in most exact models this will cause mathematical difficulties. In the airport example presented in Section 4, the number of yearly passengers from a city using the new airport most probably depend negatively on the distance.

It should also be considered what kind of pull objective (cost function) is appropriate. We have only considered the minisum. It should also be noted that for some objectives an exact bound, or at least an improved bound, may be applied. This would be the case for the maxisum objective in the planar case.

The data on hand may determine which model is the most appropriate, in a given application.

The output of the models reveal the trade-off between the two negatively correlated criteria. We conclude that the two proposed models are good tools for obnoxious location decisions.

Finally, we have illustrated the models on a real-life application.

References

[1] J. Brimberg and H. Juel. A bicriteria model for locating a semi-desirable facility in the plane. *European Journal of Operational Research*, 106:144–151, 1998.

[2] J. Brimberg and H. Juel. On locating a semi-desirable facility on the continuous plane. *International Transactions in Operational Research*, 5:59–66, 1998.

[3] J. Brumbaugh-Smith and D. Shier. An empirical investigation of some bicriterion shortest path algorithms. *European Journal of Operational Research*, 43:216–224, 1989.

[4] E. Carrizosa, E. Conde, and D. Romero-Morales. Location of a semiobnoxious facility. A biobjective approach. In 1996 Torremolinos, editor, *Advances in multiple objective and goal programming*, pages 338–346. Springer-Verlag, Berlin-Heidelberg, 1997.

[5] R. L. Church and R. S. Garfinkel. Locating an obnoxious facility on a network. *Transportation Science*, 12:107–118, 1978.

[6] E. Erkut and S. Neuman. Analytical models for locating undesirable facilities. *European Journal of Operational Research*, 40:275–291, 1989.

[7] J. R. Evans and E. Minieka. *Optimization Algorithms for Networks and Graphs.* Dekker, New York, 1992.

[8] R. L. Francis, L. F. McGinnis, and J. A. White. *Facility layout and location: An analytical approach.* Prentice Hall, New Jersey, 1992.

[9] H. W. Hamacher, M. Labbe, and S. Nickel. Multicriteria network location problems with sum objectives. *Networks*, 33:79–92, 1999.

[10] H. W. Hamacher and S. Nickel. Multicriteria planar location problems. *European Journal of Operational Research*, 94:66–86, 1996.

[11] H. W. Hamacher, S. Nickel, and A. J. V. Skriver. Multicriteria semi-obnoxious network location problems with sum and center objectives. *Working paper*, 1999. Department of Operations Research, University of Aarhus, Denmark.

[12] P. Hansen, D. Peeters, D. Richard, and J. F. Thisse. The minisum and minimax location problems revisited. *Operations Research*, 33:1251–1265, 1985.

[13] P. Hansen, D. Peeters, and J. F. Thisse. On the location of an obnoxious facility. *Sistemi Urbani*, 3:299–317, 1981.

[14] R. F. Love, J. G. Morris, and G. O. Wesolowsky. *Facilities Location : Models & Methods.* North-Holland, New York, 1988.

[15] A. J. V. Skriver and K. A. Andersen. A label correcting approach for solving bicriterion shortest path problems. *Computers and Operations Research*, 27:507–524, 2000.

[16] R. E. Steuer. *Multiple criteria optimization: Theory, Computation, and Application.* Wiley, New York, 1986.

[17] K. Thulasiraman and M. N. S. Swamy. *Graphs: Theory and Algorithms.* Wiley, New York, 1992.