Determining an Optimal Visualization of Physically Realizable Symbol Maps

Guilherme Kunigami, Pedro J. de Rezende, Cid C. de Souza
Institute of Computing
UNICAMP
Campinas, Brazil
Email: kunigami@gmail.com, {rezende, cid}@ic.unicamp.br

Tallys Yunes
School of Business Administration
University of Miami
Miami, USA
Email: tallys@miami.edu

***Abstract*—Proportional symbol maps are an often used tool to aid cartographers and geo-science professionals to visualize data associated with events (e.g., earthquakes) or geo-positioned statistical data (e.g., population). At specific locations, symbols are placed and scaled so that their areas become proportional to the magnitudes of the events or data. Recent work approaches the problem of drawing these symbols algorithmically and defines metrics to be optimized to attain different kinds of drawings. We focus specifically on optimizing the visualization of physically realizable drawings of opaque disks by maximizing the sum of the visible borders of such disks. As this problem has been proven to be NP-hard, we provide an integer programming model for its solution along with decomposition techniques designed to decrease the size of input instances. We present computational experiments to assess the performance of our model as well as the effectiveness of our decomposition techniques.**

***Keywords*—Visualization; cartography; computational geometry; integer linear programming**

I. INTRODUCTION

Proportional symbol maps are a cartographic tool employed in the visualization of geo-positioned data or events associated with locations. In these maps, symbols are placed over the points that correspond to the positions where data were gathered or events occurred, and the area of these symbols are made proportional to the magnitude of the phenomenon they represent. Commonly represented data include earthquakes (with location and intensity), and demographic statistics. While symbol shapes may vary according to applications, disks are often a very intuitive form of conveying information on the magnitude of events, so, in this paper we restrict ourselves to the placement of opaque circles. Obviously, due to the proximity of the disks and their sizes, overlapping may occur, as depicted in Fig. 1.

Depending on the scaling factor applied to the symbols, the amount of overlapping can differ greatly. Although the general rule for choosing the representation scale, as stated by Slocum et al. [1]: "neither too full nor too empty," is rather subjective, it is expected that any visually pleasing map will contain at least some overlap of symbols. Depending on the (partial) order in which the disks are organized, different portions of the symbols will be visible. The question we address here is how should a given set of disks be arranged so that the final map contains the best visual information possible.

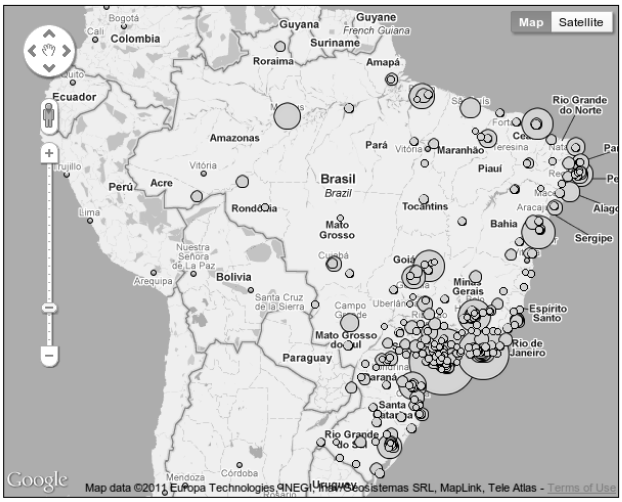

Fig. 1. A screen shot from an interface that generates proportional symbol maps from a data set.

The first paper to address this problem algorithmically was by Cabello et al. [2]. They introduced two metrics to quantify the quality of a drawing and also two possible drawings of disks, leading to four very interesting and related problems. We will now briefly describe them.

A. Notation and definitions

Employing the same definitions and notations as in [2], let S be a set of n disks in the plane. An arrangement $\mathcal{A}$ of the boundaries of these disks partitions the plane into connected regions. A vertex of $\mathcal{A}$ is the intersection point of two or more boundaries. An arc of $\mathcal{A}$ is a maximal connected portion of the boundary that connects two vertices and contains no vertex in its interior. A face of $\mathcal{A}$ is a maximal connected region bounded by arcs that does not contain any vertices or arcs in its interior. A *drawing* $\mathcal{D}$ of S is a subset of arcs and vertices of $\mathcal{A}$, denoted by $\mathcal{A}(\mathcal{D})$, drawn on top of the filled interiors of disks in S. Fig. 2 shows an arrangement and a drawing.

B. Physically Realizable Drawing

A physically realizable drawing can be thought of as a drawing constructed from whole symbols cut out from sheets of paper. Seen in this fashion, we can interleave them as in

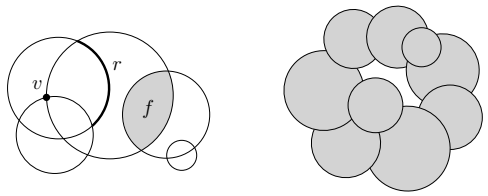

Fig. 2. An arrangement with vertex v, arc r, and face f (left), and a physically realizable drawing with interleaving disks (right).

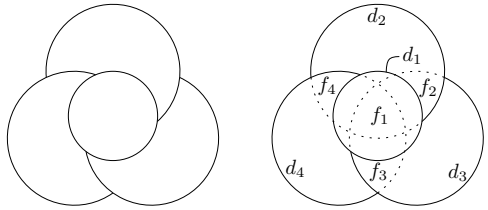

Fig. 3. A drawing that is not physically realizable and the underlying arrangement. If we remove the topmost disk, how can the remaining ones be arranged?

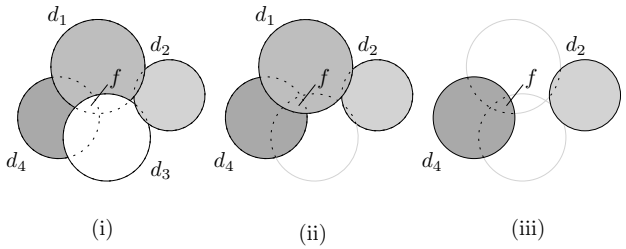

Fig. 4. A physically realizable drawing of fours disks. The region of face f from the arrangement that is initially seen in (i) belongs to d_3. After removing d_3, it belongs to d_1 in (ii), and after removing d_1, it belongs to d_4 in (iii).

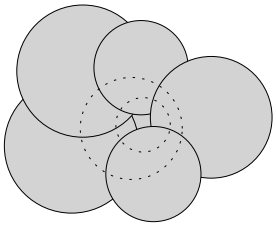

Fig. 5. An example of a disk whose border is completely covered. Its center and radius cannot be determined.

Fig. 2 (right), provided that physical restrictions are observed. For example, the drawing in Fig. 3 cannot be created without cutting the disks.

Consider the arrangement of four disks in Fig. 4(i). If the disks are of different colors, it is pretty clear that in any physically realizable drawing each face of the arrangement contained in at least one disk will have a unique color. By not allowing the disks to be cut or folded, we have that given two intersecting disks d_i and d_j, either d_i is over d_j (denoted "$d_i > d_j$"), or vice-versa. Thus, the color that is seen on a face f corresponds to that of the disk that is placed over all other disks that contain f. The iterated removal of the topmost disk, and the corresponding change in the color of f, induces a total ordering of the disks containing f.

However, this ordering alone is not enough to define a physically realizable drawing since, for instance, in the drawing in Fig. 3 there exists an order inducing sequence of colors for each face, but such drawing cannot be physically constructed. To see where the difficulty lies, notice that face f_1 induces a total order between d_1, d_2, d_3 and d_4, but any such order will conflict with the orders $d_2 > d_3$, $d_3 > d_4$ and $d_4 > d_2$ induced by faces f_2, f_3 and f_4, respectively. In other words, even though physically realizable drawings do not require a total order, multiple partial orderings of the disks engendered by the faces must not contradict each other.

On the other hand, when the requirement of total order between the disks of S is added, that is, all cyclic orders become forbidden, we have a special case of a physically realizable drawing called *stacking drawing*.

As far as arcs are concerned, an arc r from the arrangement will belong to a drawing (i.e., it will be visible) whenever there is no disk covering it. In other words, the disk of which r is border must be above all disks that contain it in their interior.

C. Visual quality of a Drawing

According to Cabello et al. [2], a good drawing should enable the viewer to see at least some part of every disk and to gauge their sizes as correctly as possible. As supporting evidence for this argument, consider Fig. 5, in which we can determine neither the center coordinates nor the radius of the hidden disk. This led to the definition of two metrics used to quantify the quality of a drawing $\mathcal{D}$ of a set S of disks. Let b_i be the total length of the visible boundary of disk $i \in S$ in $\mathcal{D}$. The *Max-Min*($\mathcal{D}$) metric is defined as $\min\{b_i | i \in S\}$. The second metric, *Max-Total*($\mathcal{D}$), provides the sum of the lengths of all visible boundaries over all disks in S.

Based on these metrics, four problems can be stated: maximizing the Max-Min (or the Max-Total) metric on a physically realizable (or a stacking drawing), depending on what type of drawings are accepted as solutions.

D. Related work

Cabello et al. [2] showed that both Max-Min and Max-Total problems for physically realizable drawings are NP-hard. On the other hand, an $O(n^2 \log n)$ algorithm for the Max-Min problem restricted to stacking drawings was presented. The complexity of the Max-Total stacking drawing problem remains open.

In [3], the authors present an integer linear programming (ILP) formulation for the Max-Total stacking drawing problem together with a theoretical study of that formulation. Since the difficulty of finding optimal solutions grows with the cardinality of the set of disks, the importance of techniques for decomposing a given instance into smaller independent ones is easily perceived. In [3], two effective methods for breaking large input sets into more manageable ones were also introduced. Furthermore, in [4] a tighter ILP formulation for the same problem was presented which turned out to be much more effective in solving the same set of instances.

E. Our Contributions

We present the first exact algorithm for the Max-Total physically realizable drawing problem. Since [4] deals with a restricted version of this problem, namely, the Max-Total stacking drawing, it is natural that some results therein will be helpful in establishing the effectiveness of our algorithm.

We provide an ILP formulation as well as a theoretical study of the resulting model. We then describe successful methods to break input instances into smaller components that can be solved separately and how to combine optimal solutions for these components to construct the optimal solution to the original instance.

With this approach, we were able to find the optimal solutions for several data sets for which only approximate ones were known and also for a number of very hard instances of statistical population-based data.

To the best of our knowledge, this is the first time that provably optimal solutions to the Max-Total physically realizable drawing problem are attained.

F. Methodology

We design an ILP formulation for the Max-Total physically realizable problem, which consists in defining a set of integer variables and an objective function to be optimized (maximized, in our case). These variables must satisfy a set of constraints, more specifically, a set of linear inequalities. In general, we seek solutions to the problem among points that satisfy all the constraints. However, in some cases we can relax the model to accept solutions that are not necessarily feasible, if we can prove the feasibility of the optimal solution. This is the case for the formulation presented here. Our model may then be solved by a branch-and-bound method.

In many cases, to try to obtain a solution restricted to the inequalities given in the description of the model leads to an algorithm of poor performance. One useful technique is to complement the model with additional constraints in the hope that they reduce the search space. In general, the number of such constraints can be too big, so only a well chosen subset of them is added. Also, this is done on demand throughout the execution of the algorithm. However, for our problem their number is small enough, so that we can incorporate them in our model before the algorithm starts. Finding additional inequalities that effectively improve the performance of the branch-and-bound algorithm is a key step in an integer programming approach. As we will see in the next section, if we can prove that a constraint satisfies certain properties, its chances of being effective in practice are much higher.

Another very useful technique is to preprocess the input data of the problem in order to improve the running time of the algorithm. In section IV, we describe the decomposition techniques presented in [4] and applicable here, as well as an extra one developed specifically for the Max-Total physically realizable drawing problem, all of which are quite effective in speeding up the algorithm.

G. Organization

In Section II, we provide a brief background on ILP. In Section III, we describe the optimization models for both stacking and physically realizable drawings, including extra inequalities that can be added to strengthen these models. Details of our implementation and experiments appear in Sections V and VI, respectively. Section VII contains an analysis of our results, followed by our final conclusions in Section VIII.

II. Integer Linear Programming Background

In this section, we briefly revise basic polyhedral combinatorics concepts necessary to understand our results. Wolsey's Integer Programming book [5] is an excellent reference for further reading on this subject.

Given an ILP model, to relax the variables in a given constraint means to allow for real values that satisfy it. This relaxation amounts to regarding that constraint as a real half-space. Therefore, in the relaxed model, a set of constraints represents the intersection of half-spaces; hence, a polyhedron.

A linear objective function represents a family of parallel hyperplanes, and maximizing this function, subject to the constraints, is the same as finding a member of this family that intercepts the polyhedron and maximizes the value of the function. If the polyhedron is not degenerated, it can be shown that at least one of its vertices represents an optimal solution to the corresponding problem. Moreover, there exist polynomial time algorithms to find such a vertex [5].

However, if we restrict the domain of the variables to the set of integers, this problem, in general, becomes NP-hard.

Let P be the set of integer points satisfying the constraints of the ILP model and consider the convex hull, conv(P), of P. Note that an optimal solution to the ILP is to be found on a vertex of conv(P).

Therefore, if we could find a polynomial description of conv(P), we might ignore the integrality constraint of the variables and solve the model in time polynomial on the size of the constraints sufficient to describe conv(P).

The difficulty here lies in that, in general, the number of inequalities to define this convex hull is exponential in the size of the input. To circumvent this obstacle, in practice, what one does is to use a *valid formulation* which is a formulation that contains all points in P and no other integer points. The model corresponding to these formulations with variables restricted to integers is then solved with a branch-and-bound algorithm.

These algorithms have theoretical exponential running times but tend to behave well in practice. Clearly, there are infinitely many valid formulations for a given set P, and those that are closer to the convex hull lead to better running times.

With this in mind, one seeks concise families of valid inequalities for the problem. Since there may exist an exponential, or even an infinite number of them, some selection process needs to come into play. For this, we use *cutting plane algorithms* which consist of solving in polynomial time a linear relaxation of the model (that is, a model without the integrality constraint on the variables) to find an optimal solution s^*. If this solution is solely comprised of integer values, the problem

is solved. Otherwise, we solve the *separation problem* which, given a family of valid inequalities, consists of (i) finding one that cuts off the optimal solution s^*, and (ii) adding it to the formulation. This procedure is repeated a specified number of times or until no such inequalities can be found. This process produces to a much stronger formulation, which should be solvable much faster by a branch-and-bound algorithm.

When removing a solution with a cutting plane, we would like to cut out as large a slice as possible from the current polyhedron in order to come closer to the convex hull. The best possible cutting planes are those that define the facets of the polyhedron, that is, those which have dimension one less than the convex hull. Therefore, when finding a new family of inequalities for a problem it is important to prove whether they are facet-defining for the convex hull of the feasible integer solutions. If the polyhedron has full dimension, that is, it has the same dimension as the space defined by the variables, proving whether inequalities are facet-defining is much easier. Thus, we always seek to work with full dimensional polyhedra in the theoretical study of these problems.

A. Usual approach

When addressing a problem through an ILP approach, one generally describes a valid ILP model and, in the theoretical side of the study, starts by determining the dimension of the polyhedron defined by the constraints of the model. Next, one verifies whether the constraints of the model define facets.

Afterward, we seek additional families of inequalities, preferably facet-defining, to use as cutting planes. It is also necessary to develop algorithms to solve the separation problem, but since it may be NP-hard, heuristics are often used.

Lastly, one must opt for an ILP solver, among the several commercial or free ones available, as this election determines the format for the provision of the constraints to the solver and, possibly, a choice of callbacks that will need to be activated whenever an optimal solution to the linear relaxation is found. These callbacks are supplied along with algorithms to solve the separation problem and to add new constraints to the model.

In the next section, we describe our ILP formulation for the Max-Total physically realizable drawing problem.

III. Integer Linear Programming Formulations

We begin by describing the model for the Max-Total stacking drawing problem presented in [4] because our model is related to it.

A. Max-Total Stacking Drawing

We need the following data, which can be calculated in polynomial time given the input set of disks S:

- $R \equiv$ set of all arcs of the arrangement;
- $\ell_r \equiv$ length of arc $r \in R$;
- $d_r \equiv$ disk that contains arc r in its border;
- $S_r^I \equiv$ set of disks that contain arc r in their interior.

The model uses two set of variables. For each arc $r \in R$, let the binary variable x_r be equal to 1 if arc r is visible, and equal to 0 otherwise. The Max-Total problem targets maximizing

$$\sum_{r \in R} \ell_r x_r \quad . \tag{1}$$

For each pair of disks $i, j \in S$, we define the binary variable w_{ij} which is equal to 1 if disk i is above disk j, and equal to 0 otherwise. The constraints are given by:

$$w_{ij} + w_{ji} \leq 1, \qquad \forall\, i, j \in S,\ i < j \tag{2}$$

$$x_r \leq w_{d_r j}, \qquad \forall\, r \in R,\ j \in S_r^I \tag{3}$$

$$w_{ij} + w_{jk} - w_{ik} \leq 1, \qquad \forall\, i, j, k \in S,\ i \neq j \neq k \neq i \tag{4}$$

$$x_r \in \{0, 1\}, \qquad \forall\, r \in R \tag{5}$$

$$w_{ij} \in \{0, 1\}, \qquad \forall\, i, j \in S,\ i \neq j \tag{6}$$

Constraint (2) states that either i is above j, or vice-versa. Constraint (3) states that if arc r is visible, its disk d_r has to be above all other disks that contain r in their interior. Finally, (4) makes sure that the (partial) order imposed by the w_{ij} variables is transitive.

In [4], the authors study the convex hull of the feasible integer solutions to (2)–(6), which we will denote by P_{SD}, from the point of view of polyhedral combinatorics. They start by proving that P_{SD} is fully dimensional, and that (2)–(4) define facets for P_{SD}. Note that for each $r \in R$, (5) represents the constraints $0 \leq x_r \leq 1$, as well as, for two distinct disks $i, j \in S$, (6) spans $0 \leq w_{ij} \leq 1$. Next, it is proven that $x \geq 0$ defines a facet of P_{SD}, while $x_r \leq 1$ does so only when $S_r^I = \emptyset$. Finally, it is also shown that $w_{ij} \geq 0$ defines a facet for P_{SD} only if $S_r^I = \emptyset$ for all r in the border of disk i, whereas $w_{ij} \leq 1$ does not define a facet of P_{SD}.

Additional Inequalities: In [3], [4], additional inequalities are introduced to strengthen the models, improving the running time of the algorithm. One of those inequalities is based on the fact that some arcs may not be simultaneously visible.

We define a graph $G_I = (V, E)$, with a vertex $v(r) \in V$ corresponding to arc $r \in R$ and an edge $(v(r_1), v(r_2))$ if d_{r_1} contains r_2 and d_{r_2} contains r_1. Two arcs whose vertices are adjacent in G_I cannot both be visible. We can extend this observation to cliques. Given a maximal clique K in G_I, we denote by $R(K)$ the set of arcs with corresponding vertices in K. We then have the following valid constraint:

$$\sum_{r \in R(K)} x_r \leq 1 \qquad \forall K \in G_I \tag{7}$$

We define a vertex of the arrangement to be non-degenerated if it is formed by the intersection of exactly two circumferences. Given one such vertex v, its neighborhood consists of four incident arcs as in Fig. 6(i). It is easy to verify that from all 16 possible configurations of visible arcs, there are only five that are actually feasible. These cases are shown in Fig. 6(ii)–(vi). Together with (7), a set of valid constraints introduced in [3] avoids all infeasible cases. Referring to r_1, r_2, r_3 and r_4 as in Fig. 6(i), those constraints are written as follows:

$$x_{r_1} \geq x_{r_3} \tag{8}$$

$$x_{r_2} \geq x_{r_4} \tag{9}$$

$$x_{r_3} + x_{r_4} \geq x_{r_1} \tag{10}$$

$$x_{r_3} + x_{r_4} \geq x_{r_2} \tag{11}$$

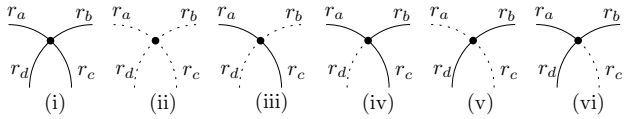

Fig. 6. Arcs incident to a non-degenerated vertex v (represented by the black dot) in (i), and feasible configurations in (ii)–(vi).

In [3], it is shown that constraints (7)-(11) are facet-defining for P_{SD}.

B. Max-Total Physically Realizable Drawing

In this section, we will show that an ILP formulation corresponding to a given subset of constraints from the previous model is valid for the Max-Total physically realizable drawing problem. Let F be the set of faces from the arrangement. Given $f \in F$, let S_f be the set of disks that contain face f.

Our model is similar to the previous one except that constraint (4) is replaced by the following:

$$w_{ij} + w_{jk} - w_{ik} \leq 1, \forall\ f \in F,\ i,j,k \in S_f\ . \qquad (12)$$

Intuitively, physically realizable drawings cannot contain all transitivity constraints in (4) because that would preclude valid drawings such as the one depicted on the right side of Fig. 2. However, the definition of a physically realizable drawing implies that transitivity needs to be enforced on disks that intersect to form a face, which gives rise to (12).

Let $\mathcal{F}_{PR}$ be the formulation with constraints (2), (3), (12), (5) and (6). Given a solution satisfying $\mathcal{F}_{PR}$ we can build a solution satisfying (2) as equality, as stated in Proposition 1. Note that if we think of the w variables as relations between disks, solutions satisfying $\mathcal{F}_{PR}$ represent partial orders between the disks of S_f for each $f \in F$. Proposition 1 shows that we can transform them into total orders without decreasing the objective value of the solution.

Proposition 1. *Given a solution satisfying $\mathcal{F}_{PR}$, we can build a solution also satisfying* (2) *as equality with greater or equal objective value.*

Proof: First, let us restrict ourselves to S_f for each $f \in F$. Define a digraph $K_{S_f} = (V, A)$, with vertex set corresponding to the disks in S_f, where $v(d)$ is the vertex correlated to disk d. There is an arc $(v(i), v(j))$ in A iff $w_{ij} = 1$. Clearly, this graph is acyclic. Let $h(v)$ be the position of vertex v in some topological order of K_{S_f}. Then, for each pair of disks $i, j \in S_f$ such that $w_{ij} = 0$ and $h(v(i)) > h(v(j))$, we set $w_{ij} = 1$ and add $(v(i), v(j))$ to A. One can see that K_{S_f} remains acyclic and thus w satisfies (12). We also now have that either $w_{ij} = 1$ or $w_{ji} = 1$ for all pair of disks $i, j \in S_f$. It is clear that for each pair $i, j \in S$, every w_{ij} or w_{ji} will be set to 1, except for those pairs such that both w_{ij} and w_{ji} were initially set to 0 and $\{i, j\} \notin S_f$ for any $f \in F$. But for these pairs, we may arbitrarily set any of them to 1, since this will not violate any transitivity constraint. Hence, this new solution satisfies (2) as equality. Because no w variable was set to 0 during this process, no x variable has decreased in value (see (3)). Therefore, the objective function value cannot go down. ■

We now define an alternative formulation $\mathcal{F}'_{PR}$ that is equivalent to $\mathcal{F}_{PR}$ with (2) replaced by an equality. Proposition 2 shows that we can solve the Max-Total physically realizable drawing problem by solving the ILP model consisting of maximizing (1) subject to $\mathcal{F}'_{PR}$.

Proposition 2. *The solution that maximizes* (1) *subject to $\mathcal{F}'_{PR}$ is an optimal solution for the Max-Total physically realizable drawing problem.*

Proof: It suffices to show that a solution that satisfies $\mathcal{F}'_{PR}$ and maximizes (1) corresponds to a physically realizable drawing and that, conversely, any physically realizable drawing corresponds to a solution that satisfies $\mathcal{F}'_{PR}$.

Let (x^*, w^*) be a solution satisfying $\mathcal{F}_{PR}$ that maximizes (1). We first note that for each $f \in F$, there exists a total order between disks in S_f induced by w^* because this relationship between disks in S_f is anti-symmetric and total, due to (2), and transitive, due to (12). Also, no two disks i, j can have their relative order differ across distinct faces because, otherwise, we would have $w^*_{ij} = w^*_{ji} = 1$, contradicting the fact that w^* is anti-symmetric. Hence, the orders induced by faces do not conflict, and it is physically possible to draw the disks following those orders. Moreover, because we are maximizing (1), any visible arc in such a drawing has its x_r set to 1. If an arc r is not visible in the drawing, there exists a disk j that contains it and is above d_r, so $w^*_{drj} = 0$ and $x^*_r = 0$.

Conversely, given a physically realizable drawing, we consider the total order induced by each face f from the arrangement. Given two disks $i, j \in S_f$, we assume, w.l.o.g., that i is above j. We then set $w_{ij} = 1$ and $w_{ji} = 0$ which clearly satisfies (2) as equality. For any three disks $i, j, k \in S_f$, assuming i is above j and j is above k, it is true that i is above k and thus such construction satisfies (12) for f. For pairs of disks i, j that do not both belong to any S_f, we arbitrarily set $w_{ij} = 1$ and $w_{ji} = 0$. Since in this case w_{ij} does not appear in (12), it is enough to observe that it satisfies (2) as equality. Given any visible arc r from this drawing, its disk d_r must be above all disks containing r, so setting $x_r = 1$ will satisfy (3). ■

C. Additional Inequalities

In a physically realizable drawing, it is also the case that, for two given arcs r_1 and r_2, if d_{r_1} contains r_2 and d_{r_2} contains r_1, then at most one of these arcs is visible. This allows us to build the graph G_I, defined for the Max-Total stacking drawing problem, and thus constraint (7) is valid for our model as well.

Going back to Fig. 6, we observe that for a non-degenerated vertex v, there is a face that is contained in the same disks as v. This means that there is a total order among such disks and therefore, locally around v, the behavior of the disks is a stacking. Because the possible configurations around v are the same for stacking and physically realizable drawings, constraints (8) through (11) are also valid for our model.

D. Polyhedral Properties

Propositions 1 and 2 allow us to maximize (1) subject to $\mathcal{F}_{PR}$, instead of $\mathcal{F}'_{PR}$, to obtain the optimal solution to the

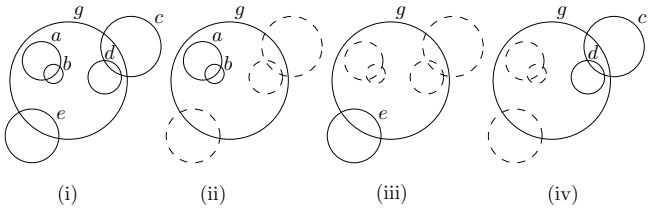

Fig. 7. An instance that allows for decomposition.

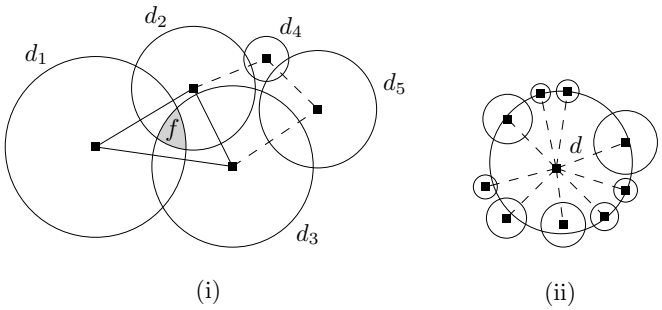

Fig. 8. (i) An instance with its corresponding disk graph. Dashed edges may be ignored in the ILP model. (ii) An instance where disk d would be replicated nine times when applying the decomposition that removes articulation points.

Max-Total physically realizable drawing problem. We denote the convex hull of the points satisfying $\mathcal{F}_{PR}$ by P_{PR}, and establish some theoretical properties of polyhedron P_{PR}.

Because $\mathcal{F}_{PR}$ contains a subset of the constraints that define polyhedron P_{SD}, the dimension of P_{PR} must be greater than or equal to the dimension of P_{SD}. Since the latter has full dimension [4], P_{PR} must have full dimension as well. Therefore, any inequality that is facet-defining for P_{SD} and is valid for P_{PR} is also facet-defining for P_{PR}. Thus, we conclude that (2), (3), (12), and (7)–(11) define facets of P_{PR}.

IV. Decomposition Techniques

In addition to the trivial decomposition that considers disjoint sets of disks independently, we argue that the two decomposition techniques presented in [3] are also valid for physically realizable drawings. Observe that if a disk d_1 is contained inside another disk d_2, there exists an optimal solution in which d_1 is drawn above d_2. In general, if two sets of disks do not intersect at their boundaries, such as sets $\{a, b\}$ and $\{c, d, e, g\}$ in Fig. 7(i), the drawing problem can be solved independently for each set. To combine those solutions, sets of disks contained inside other disks (e.g. $\{a, b\}$ are inside g) can be drawn above the disks containing them, while keeping the orders resulting from the independent solutions.

Given a set of disks S, we can define a disk graph, $G_S = (V, E)$, with a vertex $v(d) \in V$ corresponding to a disk $d \in S$ and an edge $(v(d_1), v(d_2))$ belonging to E if disks d_1 and d_2 overlap. If this graph is not biconnected, then there must exist a articulation point $v(d^*)$ in it. The removal of the corresponding disk d^* from S will spawn new connected components in G_S. It is a simple exercise to verify that if we replicate d^* in each set of disks corresponding to these components, then these augmented sets may be solved separately and their solutions easily assembled. In our example, this corresponds to the instances in Figs. 7(ii)–(iv).

We now introduce a new decomposition technique that takes advantage of the specific structure of physically realizable drawings. If a pair of disks i and j have the property that any face of the arrangement contained in both of them is not contained in any other disk, then all induced orders that include i and j are restricted to these two disks. Hence, any order we choose between i and j will not conflict with any other induced order. This establishes that we may remove the corresponding edge $(v(i), v(j))$ from G_S, solve for the connected components of G_S independently, and later decide the relative order between disks i and j, in a greedy way. For example, Fig. 8(i) depicts a set of disks with the underlying disk graph. The dashed edges can be removed from G_S. On the other hand, because face f is contained in d_1, d_2 and d_3, no edges between vertices corresponding to those disks may be removed. The resulting connected components are $\{d_1, d_2, d_3\}$, $\{d_4\}$ and $\{d_5\}$.

V. Implementation Details

A. Solver

Our implementation was done in C++ (gcc 4.4.3) and employed CGAL [6] (v3.5.1) to assemble the necessary input data for our model. We also made use of the commercial ILP solver XPRESS [7] (v20.00.05) to solve the optimization models. The experiments were run on an Intel Core 2 Quad 2.83GHz machine with 8GB of RAM, running Linux (v2.6.32).

B. ILP Model

Our original optimization model includes (2) as an equality, (3), (5), (6), and (12).

When it comes to (7), because the number of maximal cliques in a graph may be an exponential, we decided to select only some of them using the following heuristic. For each face f, let B_f^+ initially be the set of arcs r that belong to the boundary of f and whose disks d_r contain f. Let C_f^+ initially be the set of all disks that contain an arc in B_f^+. It is easy to see that the vertices corresponding to disks in C_f^+ form a clique in G_I. Since this clique is not necessarily maximal, we might try to extend C_f^+ (and its corresponding clique). Let r' be an arc contained in all disks in C_f^+ and whose disk $d_{r'}$ contains all arcs in B_f^+. The vertex set corresponding to $C_f^+ \cup \{d_{r'}\}$ forms a clique in G_I. We thus add $d_{r'}$ to C_f^+ and r' to B_f^+, and repeat this procedure until the resulting clique is maximal in G_I.

Surprisingly, our experiments showed that extending the original set C_f^+ decreases the performance of the branch-and-bound algorithm. One possible explanation is that this extension increases the density of the model (in terms of its coefficient matrix), making it harder to solve at each search node. Therefore, we opted for simply using the initial C_f^+ in the experiments reported in Section VII. As a consequence, the total number of constraints (7) and (8)–(11) is relatively small when compared to the number of constraints in the original model (16.3% on average). Hence, we decided to include all

of those constraints at the beginning of the search, instead of using a separation procedure to add them gradually as they became violated (a practice known as *branch-and-cut*).

C. XPRESS Parameters

For reproducibility purposes, we provide here the XPRESS parameters that had their default values changed in our experiments: `XPRS_MIPRELSTOP` set to 0.0, `XPRS_MIPABSSTOP` set to 10^{-7}, `XPRS_MIPRELCUTOFF` set to 0.0, `XPRS_MIPADDCUTOFF` set to 10^{-7}, and `XPRS_MAXTIME` set to -18000. For more information on these parameters and their default values, please refer to the XPRESS-Optimizer Manual [7].

VI. Problem Instances

We assess the effectiveness of our solution approach through a series of experiments with various data sets. The following data sets first appeared in [2]:

- *City 156* and *City 538* – Populations of the 156 and 538 largest cities in the United States;
- *Earthquake-Death* – Death counts due to earthquakes around the world;
- *Earthquake-Magnitude* – Magnitudes of earthquakes around the world.

In addition to the data sets above, we created additional instances consisting of the populations of the largest cities in the following countries: Belgium, China, Denmark, Indonesia, Israel, Netherlands, Norway, Spain, United Kingdom; and in eastern United States.

In Section VII we scrutinize the outcome of our computational experiments.

VII. Results and Discussion

A. Decomposition Results

We begin by discussing the effects of the decomposition techniques on the instances used in [2], which are summarized in Table I. For simplicity, we name the decompositions as follows. Decomposition A is that which regards sets of disks with no boundary intersection independently; decomposition B is the one that keeps removing articulation points until the resulting components are biconnected; and decomposition C is the new one introduced in Section IV. We reproduced the results in [3] by decomposing the original instance with A, and further decomposing each resulting component with B. We denote this chain decomposition as AB. Using similar notation, we denote by ACB the decomposition sequence of A, followed by C, and then B. The reason to perform C before B is that some cases are decomposable by either B or C, as in Fig. 8(ii), but since B replicates vertices, increasing the total number of disks to be solved, it is best to apply C first.

The first two columns of Table I indicate the names of the instances and their original number of disks. For each decomposition, we show the size of the largest component (Max), the total number of resulting components (#), and the average component size (Avg) obtained after performing the decomposition. Note that multiplying the average number (of disks) by the number of instances will not necessarily produce the number of original disks because decomposition B replicates disks.

TABLE I
Decomposition results

		Decomposition ACB			Decomposition AB		
Instance	Disks	Max	#	Avg	Max	#	Avg
City 156	156	26	66	2.39	29	53	3.09
City 538	538	53	258	2.10	53	240	2.35
Death	573	70	355	1.62	70	333	1.77
Magnitude	491	50	116	9.92	45	116	11.22

The reductions in problem size are remarkable. For example, instances City 538 and Magnitude can now be solved by optimizing over sets of disks no larger than about a tenth of their original sizes. Even after introducing decomposition C, the largest component of most instances remained unbroken, but this decomposition did split other smaller components, decreasing the average number of disks to be solved at a time.

B. Experimental Results

In the following discussion, we focus on the challenging components from City 538, Earthquake-Death and Earthquake-Magnitude because the remaining instances/components could be solved very easily. In general, instances whose G_S graphs contain large cliques tend to be the most challenging for our algorithm.

Table II summarizes our results.

TABLE II
Results on the largest components that were solved from each original problem instance. Times are reported in seconds.

	Base	Optimal	Nodes		Time	
Component	Value	Value	PR	SD [4]	PR	SD [4]
538-1-6 (29)	21.98	44.32	1	1	3	5
538-1-0 (51)	77.37	90.08	1	1	4	19
538-24-0 (53)	18.98	65.08	177	453	14554	84308
death-2-0 (70)	725.28	1152.13	1	1	6	61
mag-6-0 (26)	217.21	579.58	1	1	4	13
mag-1-1 (39)	417.32	1128.52	1	1	13	48
mag-5-0 (81)	601.79	1914.27	1	1	14	2312
mag-1-0 (113)	581.41	3158.82	3	1	107	34306
mag-7-0 (116)	700.37	2916.17	1	1	42	25256

The first column contains the component name in the form α-β-γ (δ), where α relates to the original instance ("538" for City 538, "death" for Earthquake-Death and "mag" for Earthquake-Magnitude), β identifies the component id from decomposition A, and γ indicates the γ-th component obtained after performing decomposition B on component β. Finally, δ denotes the number of remaining disks in this component. Column *Base Value* shows the total length of the arcs that are always visible in any solution, that is, those that are not contained in any disk. The *Optimal Value* column shows the value of the optimal solution minus the base value. The last four columns show the number of search nodes and time (in seconds) required by our branch-and-bound algorithm (PR) and by the algorithm (SD) in [4].

For all tested instances, PR requires less time to obtain provably optimal solutions than SD does, sometimes by more than one order of magnitude.

TABLE III
OPTIMAL SOLUTIONS FOR POPULATION INSTANCES: PR VS. SD

Country	Base Value	Optimal Solution (PR)	Optimal Solution (SD)
Belgium (312)	5354.299813	**3127.987567**	3127.787579
China (141)	1988.466543	**2409.172764**	2409.150757
Denmark (310)	4640.934381	**2301.315222**	2301.088704
Indonesia (150)	2062.608287	**1275.764650**	1275.749352
Israel (150)	1772.046049	**1892.880942**	1892.823147
Netherlands (367)	6459.025937	**4720.454497**	4720.453335
Norway (150)	2108.152483	**1230.658095**	1230.632434
Spain (300)	4469.564331	**3861.170977**	3861.156866
United Kingdom (186)	2530.147579	**1999.491671**	1999.490668
United States (East) (87)	1810.833802	**2462.579637**	2462.556433

The results for population-based instances are split between Tables III and IV. We selected ten instances for which an optimal physically realizable drawing has an objective value strictly greater than the value of an optimal stacking drawing, thus resulting in visually better solutions. See Fig. 9 for an example.

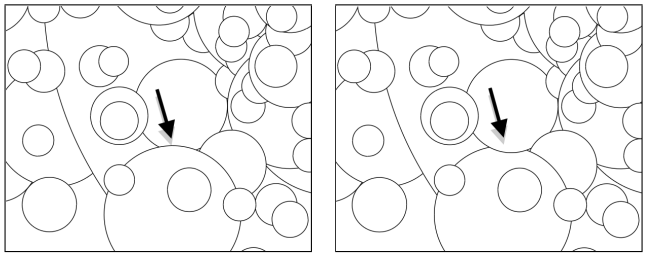

Fig. 9. A close up of the Denmark instance, showing a difference between the SD and PR optimal solutions.

Table III shows the base values as well as the optimal solution values obtained by the physically realizable (PR) and the stacking drawing (SD) algorithms, respectively. Table IV complements Table III with the number of nodes explored by each algorithm and their execution times.

TABLE IV
SEARCH NODES AND CPU TIME FOR POPULATION INSTANCES: PR VS. SD

	PR		SD	
Country	Nodes	Times	Nodes	Times
Belgium (312)	18	6	18	2757
China (141)	5	6	5	3797
Denmark (310)	19	9	19	1007
Indonesia (150)	2	14	2	2477
Israel (150)	6	22	6	1044
Netherlands (367)	23	9	24	4222
Norway (150)	4	11	3	3433
Spain (300)	16	54	15	194
United Kingdom (186)	8	7	8	422
United States (East) (87)	8	22	6	2788

Although the solution values of the physically realizable drawings in Table III are only slightly greater than their stacking counterparts, surprisingly, when looking at a map with hundreds of disks, even a minor improvement can be significant. In fact, it can mean the difference between seeing or missing a city.

In terms of execution times, once again, the PR algorithm is *greatly* superior to the SD algorithm, as shown in Table IV. As before, improvements can range from one to more than two orders of magnitude.

VIII. CONCLUSION

We propose and implement an exact algorithm to solve the NP-hard problem of generating physically realizable drawings of proportional symbol maps, which are an important visualization tool for geo-positioned data. Furthermore, we describe in detail the results of an extensive experimental study on the behavior of our method.

The symbols under consideration are opaque disks whose areas are proportional to the magnitude of the events or data they represent.

Our optimization approach is based on an integer linear programming formulation that maximizes the total length of the visible borders of the disks on the map (an established measure of quality). The importance of physically realizable drawings stems from the fact that they improve on the previously studied stacking drawings by exposing greater portions of the disk borders.

We enhance the performance of our optimization model by using known and novel decomposition techniques, as well as several families of facet-defining inequalities.

Our computational results, which involve real life data sets related to natural events and population statistics, indicate that, in addition to being visually superior, optimal physically realizable drawings can be obtained at a fraction of the computational effort required to obtain optimal stacking drawings.

To the best of our knowledge, we are the first to find provably optimal physically realizable drawings of the data sets proposed in [2], as well as of the population-based data sets described in Section VI of this paper.

ACKNOWLEDGMENT

Guilherme Kunigami is supported by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) grant 830510/1999-0. Pedro J. de Rezende is partially supported by CNPq grants 472504/2007-0, 483177/2009-1, 473867/2010-9, FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo) grant 07/52015-0, and a grant from FAEPEX/UNICAMP. Cid C. de Souza is partially supported by CNPq grants 301732/2007-8, 472504/2007-0, 473867/2010-9 and FAPESP grant 07/52015-0.

REFERENCES

[1] T. A. Slocum, R. B. McMaster, F. C. Kessler, and H. H. Howard, *Thematic cartography and geographic visualization*, 2nd ed. Prentice Hall, 2003.
[2] S. Cabello, H. Haverkort, M. van Kreveld, and B. Speckmann, "Algorithmic aspects of proportional symbol maps," *Algorithmica*, vol. 58, no. 3, pp. 543–565, 2010.
[3] G. Kunigami, P. J. de Rezende, C. C. de Souza, and T. Yunes, "Optimizing the layout of proportional symbol maps," in *Proceedings of ICCSA 2011*, ser. LNCS, B. Murgante, O. Gervasi, A. Iglesias, D. Taniar, and B. Apduhan, Eds., vol. 6784. Springer-Verlag, 2011, pp. 1–16.
[4] ——, "Optimizing the layout of proportional symbol maps: polyhedra and computation," *unpublished*, 2011.
[5] L. A. Wolsey, *Integer programming*. John Wiley and Sons, Inc., 1998.
[6] "CGAL, Computational Geometry Algorithms Library," www.cgal.org.
[7] Fair Isaac Corporation, *Xpress optimizer reference manual*, 2009.