Discrete Optimization 18 (2015) 87–110

Contents lists available at ScienceDirect

Discrete Optimization

www.elsevier.com/locate/disopt

Arc-based integer programming formulations for three variants of proportional symbol maps☆

Rafael G. Cano[a], Cid C. de Souza[a,*], Pedro J. de Rezende[a], Tallys Yunes[b]

[a] *Institute of Computing, University of Campinas, Campinas, SP 13083-852, Brazil*
[b] *School of Business Administration, University of Miami, Coral Gables, FL 33124-8237, USA*

HIGHLIGHTS

- A new formulation is described to create optimal proportional symbol maps.
- We address three problem variants, two of which are known to be NP-hard.
- Efficient separation routines and lifting procedures are described.
- The new formulation is up to 82 times faster than the others in the literature.
- Most known benchmark instances can now be solved in less than one minute.

ARTICLE INFO

Article history:
Received 14 January 2014
Received in revised form 12 February 2015
Accepted 2 September 2015

MSC:
90C10
90C27
90C57
90C90
52B12

Keywords:
Integer linear programming
Computational geometry
Symbol maps
Cartography
Visualization

ABSTRACT

Proportional symbol maps are a cartographic tool that employs scaled symbols to represent data associated with specific locations. The symbols we consider are opaque disks, which may be partially covered by other overlapping disks. We address the problem of creating a suitable drawing of the disks that maximizes one of two quality metrics: the total and the minimum visible length of disk boundaries. We study three variants of this problem, two of which are known to be NP-hard and another whose complexity is open. We propose novel integer programming formulations for each problem variant and test them on real-world instances with a branch-and-cut algorithm. When compared with state-of-the-art models from the literature, our models significantly reduce computation times for most instances.

☆ Research supported by grants from Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) #477692/2012-5, #311140/2014-9, #302804/2010-2, Fundação de Amparo à Pesquisa do Estado de São Paulo (Fapesp) #2007/52015-0, #2012/00673-2.

* Corresponding author.
E-mail addresses: rgcano@ic.unicamp.br (R.G. Cano), cid@ic.unicamp.br (C.C. de Souza), rezende@ic.unicamp.br (P.J. de Rezende), tallys@miami.edu (T. Yunes).

http://dx.doi.org/10.1016/j.disopt.2015.09.001
1572-5286/

1. Introduction

Proportional symbol maps are a cartographic tool to visualize data associated with specific locations (e.g., earthquake magnitudes and city populations). Symbols whose area is proportional to the numerical values they represent are placed at the locations where those values were collected. Although symbols can be of any geometric shape, opaque disks are the most frequently used and, for that reason, the focus of our study. Fig. 1 shows an example in which the area of each disk is proportional to the population of a city in central Europe.

Depending on the size and the position of the disks, some of them may be (partially) obscured. Although the literature contains studies on symbol sizing, it is unclear how much they should overlap (see [1,2]). When large portions of a disk are covered, it is difficult to deduce its size and the location of its center. Therefore, the order in which the disks are drawn affects the amount and quality of information that can be inferred from a symbol map.

1.1. Problem description

Let S be a set of n disks in the plane and $\mathcal{A}$ be the arrangement formed by the boundaries of the disks in S, as illustrated in Fig. 2(a). An intersection point of the boundaries of two or more disks defines a *vertex* of $\mathcal{A}$. The portion of the boundary of a disk that connects two vertices and contains no other vertices in between is called an *arc*. A maximal connected region delimited by arcs with no vertices in its interior is a *face* of $\mathcal{A}$. A *drawing* of S is a subset of the arcs and vertices of $\mathcal{A}$ that is drawn on top of the filled interiors of the disks in S. An example is shown in Fig. 2(b).

Cabello et al. [3] observed that the quality of a drawing depends on the visible boundary of the disks rather than on their visible area. As depicted in Fig. 2(c), a disk that has no visible boundary transmits little or no information, because it is not possible to determine its size or the position of its center. Based on this remark, the authors considered two quality metrics: the minimum visible boundary length of any disk and the total visible boundary length over all disks. The *Max–Min* and *Max-Total* problems consist in maximizing the former and the latter values, respectively.

Not every subset of the arcs and vertices of $\mathcal{A}$ yields a suitable drawing for use in a symbol map. Two types of drawings can be used, namely, *physically realizable drawings* and *stacking drawings*. A drawing $\mathcal{D}$ is physically realizable if, for each face f of $\mathcal{A}$, there exists a (strict) total order on the disks that contain f such that the following conditions hold.

1. An arc r on the boundary of a disk d_r is visible in $\mathcal{D}$ if and only if, for every disk j that contains r in its interior, d_r is above j (denoted interchangeably by $d_r \succ j$ or $j \prec d_r$).
2. Total orders associated with distinct faces are consistent with each other, i.e., two disks i and j stand in the same relationship to each other in all total orders associated with faces that are contained in both i and j.

Informally, this definition states that a drawing is physically realizable if it can be constructed from whole symbols, cut out from sheets of paper. The disks can be interleaved and warped, but cannot be cut. Fig. 3 shows an example of a drawing that is not physically realizable. Since arcs a, b and c are visible, in order to satisfy Condition 1 we must have $1 \succ 2, 2 \succ 3$ and $3 \succ 1$. But any total order assigned to face f will contradict one of these relationships, precluding the satisfaction of Condition 2.

It should be noted that physically realizable drawings do not necessarily exhibit a total order among all disks. This allows us to create drawings that have a somewhat cyclic structure, as the one shown in Fig. 4(a). The imposition of the need for a total order results in the second type of drawings. Stacking drawings are a restriction of physically realizable drawings in which there exists a total order relation on S, i.e., they

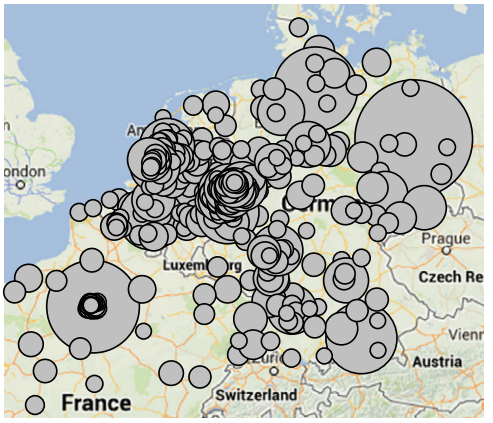

Fig. 1. Proportional symbol map showing the 300 largest cities in Germany, France, Belgium and the Netherlands.

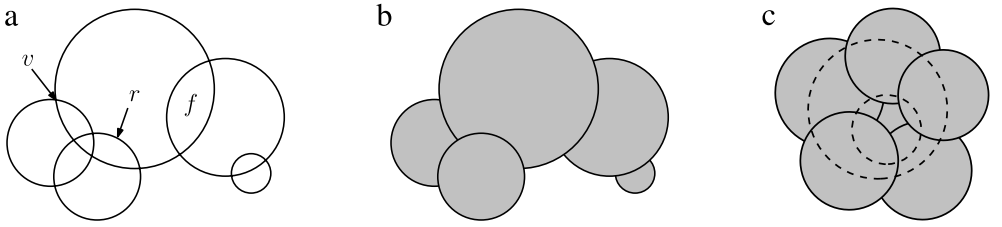

Fig. 2. (a) An arrangement $\mathcal{A}$ with vertex v, arc r, and face f; (b) a drawing of the disks in $\mathcal{A}$; (c) a drawing in which visible boundary is more important than visible area.

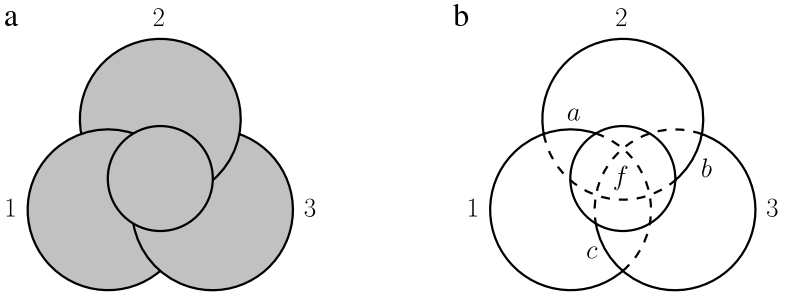

Fig. 3. (a) A drawing that is not physically realizable and (b) the underlying arrangement.

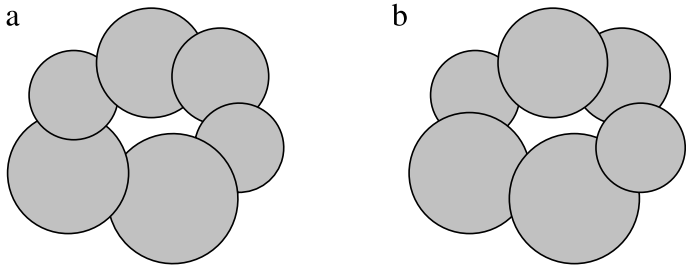

Fig. 4. (a) A physically realizable drawing that does not exhibit a total order (i.e., that is not a stacking drawing); (b) a stacking drawing.

correspond to the disks being stacked up in layers, starting with the ones on the bottom layer. An example is given in Fig. 4(b).

We can now formally state our problem: given a set S of opaque disks, construct a physically realizable drawing or a stacking drawing of S that maximizes the Max–Min or the Max-Total metric. As in [4], we refer to the Physically Realizable Drawing Problem as PRDP and to the Stacking Drawing problem as SDP.

1.2. Related work

Cabello et al. [3] are the first to study proportional symbol maps problems algorithmically. They identify and formally define the two types of drawings and the two quality metrics considered here. The authors prove that the PRDP is NP-hard for both objective functions. In addition, they describe a greedy algorithm to optimally solve the Max–Min SDP in $O(n^2 \log n)$ time. Computational results reported in that work show that this algorithm also performs relatively well as a heuristic for the Max-Total SDP. Nevertheless, in general, it does not produce optimal solutions for the latter problem, whose computational complexity remains open.

Kunigami et al. [5] propose two integer linear programming models for the Max-Total SDP. The one that performs better, which is later referred to as *Graph Orientation Model* (GOM) [6], is extended in [4] to solve both versions of PRDP. They are the first to find provably optimal solutions for these three variants. The GOM model is based on two sets of binary variables: an arc variable x_r for each $r \in R$ (to indicate whether r is visible in the solution) and an auxiliary ordering variable w_{ij} for each pair of disks $i, j \in S$ (to indicate the relative order between i and j). The authors also introduce decomposition techniques that greatly contribute to reduce the size of input instances.

Two other papers address the Max-Total SDP from different perspectives. Cano et al. [7] develop a GRASP heuristic, which is hybridized with path-relinking and variable neighborhood search. Both sequential and parallel implementations are described and experimentally evaluated. Additionally, a set of instances is presented that cannot be solved by either one of the models from [5]. In a more theoretical work, Nivasch et al. [8] provide bounds on the Max-Total value for stacking drawings of unit disks whose centers form a dense point set.

1.3. Our contribution

In this work, we address the three problem variants to which no polynomial time algorithm is known, namely, the Max-Total SDP and PRDP, and the Max–Min PRDP. Our goal is to find provably optimal solutions in reasonable time via integer linear programming (ILP). As mentioned in Section 1.2, the GOM model is an extended formulation that uses $O(n^2)$ auxiliary w_{ij} variables. In spite of having a polynomial number of constraints, the large number of extra variables increases the computation time needed to solve its linear relaxations. Therefore, lighter models are required to solve the more challenging instances, such as the ones from [7].

We propose a novel natural formulation, which is able to solve several instances that are beyond the capabilities of existing state-of-the-art exact algorithms. Since it uses only arc variables, we refer to it as *Arc Model* (AM). We prove that, for SDP, AM is a projection of GOM and, consequently, provides the same dual bounds. As it often happens with natural formulations, the new model has an exponential number of inequalities, so we also describe fast separation routines for both types of drawings. Thanks to the effectiveness of these routines and to the reduction in the number of variables, the linear relaxations of our model can be solved significantly faster.

Moreover, it is relatively straightforward to strengthen inequalities of the new formulation by applying lifting procedures. This has a substantial impact on both the quality of the dual bounds and the number of

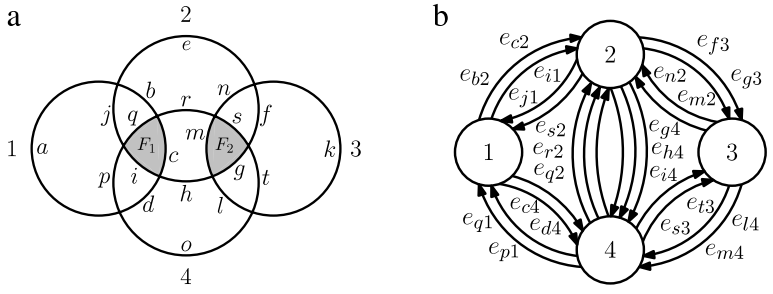

Fig. 5. (a) An arrangement with four disks and (b) its graph G_O.

nodes explored in the enumeration tree. This is an interesting development especially because none of the inequalities in the GOM model can be lifted; as shown in [5,4], they are all facet-defining. To the best of our knowledge, the improved model is currently the strongest known in the literature to deal with symbol maps problems. Computational experiments show that it is up to 82 times faster than the GOM model for some benchmark instances.

The remainder of the text is organized as follows. Section 2 describes the AM model and Section 3 contains some polyhedral results. In Section 4 we deal with the more practical issue of selecting an initial set of inequalities to start solving the problem. Separation routines and lifting procedures are presented in Sections 5 and 6, respectively. Computational results for several real-world instances are reported in Section 7. Section 8 concludes the paper and gives some directions for future research.

2. Integer linear programming model

The following notation will be used in the description of the model. Given a set of disks S and the associated arrangement $\mathcal{A}$, let R and F be the set of arcs and faces of $\mathcal{A}$. For each arc $r \in R$, we denote by ℓ_r the length of r and by d_r the disk in S whose boundary contains r. In addition, given an arc r (face f), let S_r (S_f) denote the set of disks that contain arc r (face f) in their interior.

For each arc $r \in R$, we define a binary variable x_r that is equal to 1 if r is visible in the drawing, and equal to 0 otherwise. For the Max-Total problem, the objective is to maximize $\sum_{r\in R} \ell_r x_r$. As in [4], for the Max–Min problem we must maximize an additional continuous variable z, which is added to the model together with constraints (1).

$$z \leq \sum_{r\in R\,:\,d_r=i} \ell_r x_r, \quad \forall\, i \in S. \tag{1}$$

2.1. Inequalities for stacking drawings

We begin with the formulation for stacking drawings, which is conceptually simpler. Note that when an arc r is visible in a drawing, it induces an order among d_r and every disk in S_r, i.e., d_r must be drawn above every disk that contains r. To capture all of these relations, we define an *induced order graph* $G_O = (V(G_O), E(G_O))$ as a directed multigraph with a vertex $v_i \in V(G_O)$ for each disk $i \in S$, and a directed edge $e_{rj} = (v_{d_r}, v_j)$ for each arc $r \in R$ and disk $j \in S_r$. Each edge indicates the required relative order between pairs of disks when some arc is visible. An example is shown in Fig. 5.

Let $\mathcal{C}$ be the set of all directed cycles in G_O. Given a cycle $C \in \mathcal{C}$, let R_C be the set of arcs that give rise to the edges of C, i.e., $R_C = \{r : e_{rj} \in C,\ \text{for some } j \in S\}$. If all the arcs in R_C were simultaneously visible in a solution, they would induce a cyclic order among the corresponding disks. But no (strict) total

order may contain cycles, which implies that at least one arc from R_C cannot be visible. Thus, (2) must be satisfied.

$$\sum_{r \in R_C} x_r \leq |C| - 1, \quad \forall\, C \in \mathcal{C}. \tag{2}$$

It should be noted that (2) are not only necessary but also sufficient to model the SDP constraints. To see that, suppose we are given an optimal solution x^* that satisfies (2). Let $G_O[x^*]$ denote the subgraph of G_O obtained by deleting the edges e_{rj} for which $x_r = 0$. Note that $G_O[x^*]$ is acyclic, otherwise x^* would violate one of the inequalities (2). Therefore, we can run a topological sort on $G_O[x^*]$ and use the resulting order of the vertices as the stacking order in a drawing of the disks. An arc r will be visible in this drawing if and only if $x^*_r = 1$.

2.2. Inequalities for physically realizable drawings

We now turn our attention to physically realizable drawings. Although some cycles are allowed, we still have to guarantee that total orders exist for the faces of $\mathcal{A}$ and that they are all consistent. In particular, this means that transitivity applies to (and only to) sets of disks that contain a common face. So, suppose we are given a set of arcs $T \subseteq R$ and we want to decide whether they can be visible together in a physically realizable drawing. The most direct way of doing this is to start with the relative orders induced by the arcs in T and then to enforce transitivity whenever it is applicable. Any conflicts in the orders among the disks detected during this process are an indication that the arcs in T cannot be visible simultaneously. This motivates the main definition of this section.

Let us first introduce some notation. We denote by $G_O[X]$ the subgraph of G_O induced by the set of objects X. For a set of vertices or edges, the standard definition applies. When X is a set of disks in S, $G_O[X]$ is the subgraph induced by the vertices $\{v_i : i \in X\}$. Finally, when X is a set of arcs in R, $G_O[X]$ is the subgraph induced by the edges $\{e_{rj} : r \in X\}$.

Now, let $H_O = G_O[T]$, where T is the set of arcs mentioned previously. We define the *face-transitive closure* H^*_O of H_O as its minimal supergraph such that for each face $f \in F$, $H^*_O[S_f]$ is transitively closed. The existence of a cycle in any subgraph $H^*_O[S_f]$ implies that, if the arcs in T are all visible, it is not possible to assign a total order to the disks that contain face f.

In order to describe the model, we are interested in determining which cycles are allowed in physically realizable drawings. This way, we take H_O to be a cycle $C \in \mathcal{C}$ and consider its face-transitive closure C^*. Following the previous discussion, the arcs in R_C can all be visible in a solution if and only if all subgraphs $C^*[S_f]$ are acyclic. Let $\widehat{\mathcal{C}} \subseteq \mathcal{C}$ be the set of cycles that do not satisfy this condition. For PRDP, we must replace inequalities (2) by (3).

$$\sum_{r \in R_C} x_r \leq |C| - 1, \quad \forall\, C \in \widehat{\mathcal{C}}. \tag{3}$$

Next, we show how to compute the face-transitive closure of a graph. Although the algorithm is not required for actually solving real instances, it is the basis for the separation routine that will be presented in Section 5. Initially, consider two faces $f, f' \in F$ and suppose that $S_f \subset S_{f'}$. Note that $H_O[S_f]$ is an induced subgraph of $H_O[S_{f'}]$. Thus, if the latter is transitively closed, so is the former. It follows that we only need to take a face $f \in F$ into account if it is maximal with respect to this property, i.e., if there exists no other face f' such that $S_f \subset S_{f'}$. The face-transitive closure of H_O can be obtained by repeatedly computing the transitive closure of the subgraphs $H_O[S_f]$ for each maximal face f until all of them are transitively closed.

Fig. 6 illustrates the steps to compute the face-transitive closure of a cycle C formed by arcs b, f, l, and p from the arrangement in Fig. 5(a). There are two maximal faces F_1 and F_2, and the algorithm must compute

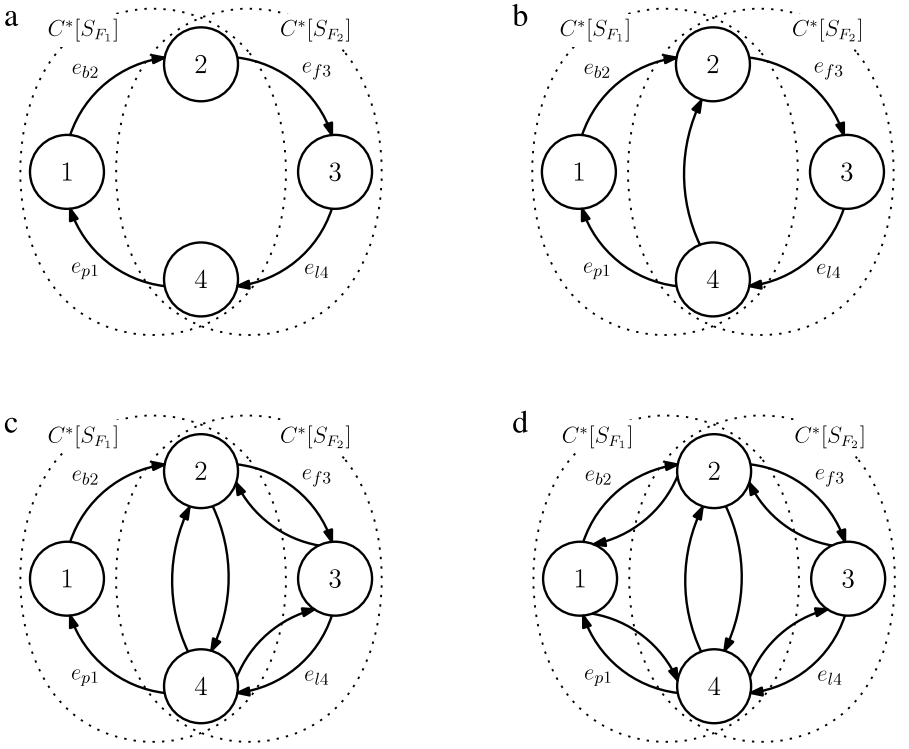

Fig. 6. (a) Cycle C formed by arcs b, f, l, and p from Fig. 5(a); result after computing the transitive closures of (b) $C[S_{F_1}]$ and (c) $C[S_{F_2}]$; (d) C^* is completed after computing the transitive closure of $C[S_{F_1}]$ for the last time.

the transitive closure of $C[S_{F_1}]$ and $C[S_{F_2}]$ until they are both transitively closed. The final result C^* shows that C is not physically realizable. In particular, this means that all feasible solutions to that arrangement are stacking drawings.

3. Polyhedral study

We now prove some polyhedral results. The following notation will be used. Given a model M, we denote by M-PRD and M-SD the specific variants designed to solve PRDP and SDP, respectively. Also, let P_M and $\widetilde{P}_M$ be the polyhedra defined by the convex hull of all integer feasible solutions of M and by the linear relaxation of M, respectively.

3.1. Relationship between AM-SD and GOM-SD

In the main theorem of this section, we prove that AM-SD is a projection of GOM-SD. Several other facial results follow directly from that fact. For ease of reference, we start by reproducing the GOM-SD model. For each $r \in R$, let the binary variable x_r be equal to 1 if r is visible in the drawing, and to 0 otherwise. Also, for every pair of distinct disks $i, j \in S$, let w_{ij} be a binary variable that is equal to 1 if i is placed above j, and to 0 otherwise. The objective for each of the problem variants is the same as in AM. For SDP, the following constraints must be satisfied:

$$x_r \leq w_{d_r j}, \quad \forall\, r \in R,\ j \in S_r \tag{4}$$

$$w_{ij} + w_{ji} \leq 1, \quad \forall\, i, j \in S,\ i < j \tag{5}$$

$$w_{ij} + w_{jk} - w_{ik} \leq 1, \quad \forall\, i, j, k \in S,\ i \neq j \neq k \neq i. \tag{6}$$

We now present two auxiliary results that will be useful for proving our propositions. Let $D = (V, E)$ be a complete directed graph. We associate a binary variable w_{uv} with each directed edge $(u, v) \in E$. The *acyclic subgraph polytope* [9], which we denote by P_{AC}, is defined as the convex hull of the integer feasible points that satisfy inequalities (7):

$$\sum_{(u,v)\in C} w_{uv} \leq |C| - 1, \quad \forall \text{ directed cycles } C \text{ in } D. \tag{7}$$

Similarly, the *partial order polytope* [10], which we denote by P_{PO}, is defined as the convex hull of the integer feasible points that satisfy inequalities (8) and (9):

$$w_{uv} + w_{vu} \leq 1, \quad \forall\, u, v \in V,\ u \neq v \tag{8}$$

$$w_{tu} + w_{uv} - w_{tv} \leq 1, \quad \forall\, t, u, v \in V,\ t \neq u \neq v \neq t. \tag{9}$$

Proposition 1. *Given $w \in \widetilde{P}_{\mathrm{AC}}$, there exists a vector $w' \geq w$ such that $w' \in \widetilde{P}_{\mathrm{AC}}$ and $w' \in \widetilde{P}_{\mathrm{PO}}$.*

Proof. We show how to transform the initial vector w into another vector w' that satisfies the conditions stated in the proposition. In particular, we will create a vector w' that satisfies inequalities (8) at equality.

Initially, note that there can be no pair of vertices u and v such that $w_{uv} + w_{vu} > 1$, otherwise the inequality (7) associated with the cycle $\{(u, v), (v, u)\}$ would be violated. Thus, w satisfies all inequalities (8). To guarantee that $w' \geq w$, we construct it by increasing the value of some elements of w. Let m be the number of inequalities (8) that are not satisfied at equality by w. We will show by induction on m that, given $w \in \widetilde{P}_{\mathrm{AC}}$, w' can always be constructed in such a way that it also belongs to $\widetilde{P}_{\mathrm{AC}}$ and satisfies (8) at equality. Inequalities (9) will be addressed later to show that $w' \in \widetilde{P}_{\mathrm{PO}}$.

For $m = 0$, we can simply set $w' := w$. For $m > 0$, there exist two vertices u and v such that $w_{uv} + w_{vu} < 1$. Among all cycles in D that contain (u, v), let $C^{\min}_{uv}$ be the one whose corresponding inequality (7) has the smallest slack, i.e., such that $s_{uv} := |C^{\min}_{uv}| - 1 - \sum_{(k,l)\in C^{\min}_{uv}} w_{kl}$ is minimum. Analogously, let $C^{\min}_{vu}$ be the cycle that contains (v, u) with the smallest slack $s_{vu} := |C^{\min}_{vu}| - 1 - \sum_{(k,l)\in C^{\min}_{vu}} w_{kl}$. Define $\delta_{uv} := \min\{s_{uv}, 1 - w_{uv} - w_{vu}\}$ and $\delta_{vu} := 1 - w_{uv} - w_{vu} - \delta_{uv}$.

Let $\widehat{w}$ be the vector obtained from w by adding δ_{uv} to w_{uv} and δ_{vu} to w_{vu}, i.e., $\widehat{w}_{uv} = w_{uv} + \delta_{uv}, \widehat{w}_{vu} = w_{vu} + \delta_{vu}$ and $\widehat{w}_{kl} = w_{kl}$ for all $\{k, l\} \neq \{u, v\}$. Note that $\widehat{w}_{uv} + \widehat{w}_{vu} = 1$, so $\widehat{w}$ still satisfies all inequalities (8); also, $m - 1$ of them are not satisfied at equality. Therefore, if we show that $\widehat{w}$ satisfies all inequalities (7), we can apply the induction hypothesis to $\widehat{w}$ and obtain the desired vector w'.

Vectors w and $\widehat{w}$ differ only in $\widehat{w}_{uv}$ and $\widehat{w}_{vu}$. Consequently, the only inequalities (7) that can be violated by $\widehat{w}$ are the ones that contain these variables. Due to our choice $C^{\min}_{uv}$ and $C^{\min}_{vu}$, it suffices to show that both of those associated with these cycles are satisfied by $\widehat{w}$. Clearly, the one associated with $C^{\min}_{uv}$ remains satisfied, so suppose the one for $C^{\min}_{vu}$ is violated. Algebraically, that means

$$\sum_{(k,l)\in C^{\min}_{vu}} \widehat{w}_{kl} = \delta_{vu} + \sum_{(k,l)\in C^{\min}_{vu}} w_{kl} > |C^{\min}_{vu}| - 1.$$

The last inequality can be rewritten as $\delta_{vu} > s_{vu} \geq 0$, which implies $\delta_{uv} = s_{uv}$ (otherwise δ_{vu} would be 0). Since $w_{uv} + w_{vu} + \delta_{uv} + \delta_{vu} = 1$, we get:

$$w_{uv} + w_{vu} + s_{uv} + s_{vu} < 1. \tag{10}$$

Consider now the two paths $\pi_{vu} := C^{\min}_{uv} - (u, v)$ and $\pi_{uv} := C^{\min}_{vu} - (v, u)$ from v to u and from u to v, respectively. As shown in Fig. 7, the union of these paths results in a closed walk, which may be decomposed

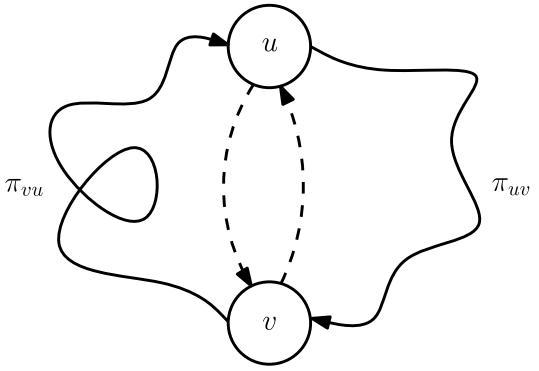

Fig. 7. Closed walk produced by the union of the paths π_{vu} and π_{uv}. These paths are not always vertex-disjoint, so $\pi_{vu} \cup \pi_{vu}$ need not be a cycle.

into a set $\mathcal{C}_{walk}$ of one or more cycles. Each one of them is associated with one cycle inequality (7) that must be satisfied by w (recall that $w \in \widetilde{P}_{\mathrm{AC}}$). Adding these inequalities yields:

$$\sum_{(k,l)\in\pi_{uv}} w_{kl} + \sum_{(k,l)\in\pi_{vu}} w_{kl} \leq |C^{\min}_{uv}| + |C^{\min}_{vu}| - |\mathcal{C}_{walk}| - 2. \tag{11}$$

But the left-hand side of inequality (11) may be rewritten as follows:

$$\begin{aligned}\sum_{(k,l)\in\pi_{uv}} w_{kl} + \sum_{(k,l)\in\pi_{vu}} w_{kl} &= \left(\sum_{(k,l)\in C^{\min}_{uv}} w_{kl}\right) + \left(\sum_{(k,l)\in C^{\min}_{vu}} w_{kl}\right) - w_{uv} - w_{vu} \\ &= |C^{\min}_{uv}| + |C^{\min}_{vu}| - w_{uv} - w_{vu} - s_{uv} - s_{vu} - 2. \end{aligned} \tag{12}$$

Using expression (12) together with inequality (10) and the fact that $|\mathcal{C}_{walk}| \geq 1$, we conclude that inequality (11) cannot be satisfied. Therefore, there is at least one inequality (7) that is violated by w, contradicting our initial assumption that $w \in \widetilde{P}_{\mathrm{AC}}$. Thus, the inequality associated with $C^{\min}_{vu}$ is also satisfied by $\widehat{w}$ and our induction proof is complete.

So far, we have proved that we may indeed construct $w' \geq w$ such that $w' \in \widetilde{P}_{\mathrm{AC}}$ and w' satisfies all inequalities (8) at equality. It remains to show that inequalities (9) are also satisfied by w'. Note that, since all inequalities (7) are satisfied by w', for any three distinct vertices $t, u, v \in V$ we must have $w'_{tu} + w'_{uv} + w'_{vt} \leq 2$. But $w'_{tv} + w'_{vt} = 1$, which leads to $w'_{tu} + w'_{uv} - w'_{tv} \leq 1$, that is the desired inequality. □

Proposition 2. $\widetilde{P}_{\mathrm{PO}} \subseteq \widetilde{P}_{\mathrm{AC}}$.

Proof. We must show that every point $w \in \widetilde{P}_{\mathrm{PO}}$ satisfies the cycle inequalities (7). Suppose that this is not true and let $w \in \widetilde{P}_{\mathrm{PO}}$ be a point that violates at least one cycle inequality. Let C be the shortest cycle in D whose corresponding inequality (7) is violated, i.e., such that

$$\sum_{(k,l)\in C} w_{kl} > |C| - 1. \tag{13}$$

Note that, since all inequalities (8) are satisfied, $|C| \geq 3$. Let t, u, v be three consecutive vertices in C. Since $w \in \widetilde{P}_{\mathrm{PO}}$, we must have

$$w_{tu} + w_{uv} - w_{tv} \leq 1. \tag{14}$$

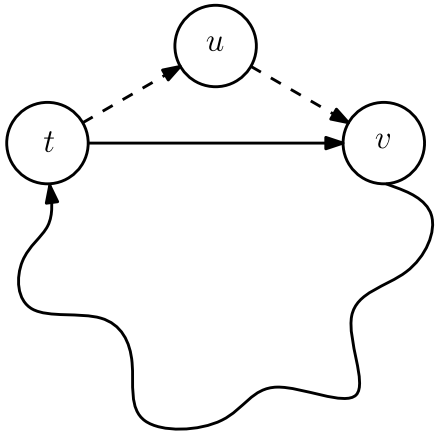

Fig. 8. Cycle C' obtained from C after replacing (t, u) and (u, v) by (t, v).

Consider now the cycle C' obtained by taking all the edges of C, except for (t, u) and (u, v) that are replaced by (t, v), as shown in Fig. 8. Using (13) and (14) we get:

$$\sum_{(k,l)\in C'} w_{kl} > |C| - 1 + w_{tv} - w_{tu} - w_{uv} \geq |C'| - 1. \tag{15}$$

This shows that the cycle inequality associated with C' is also violated, which contradicts our initial assumption and completes the proof. □

We can now prove the results stated in the main text. Given a polytope P, we denote its projection onto the x-space by $\text{Proj}_x(P)$.

Theorem 3. *The polyhedra $\widetilde{P}_{\text{AM–SD}}$ and $P_{\text{AM–SD}}$ are the projections of $\widetilde{P}_{\text{GOM–SD}}$ and $P_{\text{GOM–SD}}$ onto the x-space, respectively.*

Proof. We begin by proving that $\widetilde{P}_{\text{AM–SD}} = \text{Proj}_x(\widetilde{P}_{\text{GOM–SD}})$. It suffices to show that $x \in \widetilde{P}_{\text{AM–SD}}$ if and only if there exists a vector w such that (w, x) is a feasible point of $\widetilde{P}_{\text{GOM–SD}}$.

Suppose $x \in \widetilde{P}_{\text{AM–SD}}$. We build a vector w such that $(w, x) \in \widetilde{P}_{\text{GOM–SD}}$ as follows. For each pair of distinct disks i and j, let r be the arc with maximum x_r among the ones that are in the boundary of i and also contained in the interior of j. Each of the inequalities (4) can be satisfied by setting w_{ij} to x_r. If w violates any of the inequalities (7), there must also be an inequality (2) that is violated. However, since $x \in \widetilde{P}_{\text{AM–SD}}$, we conclude that $w \in \widetilde{P}_{\text{AC}}$. Finally, by Proposition 1, there exists $w' \geq w$ such that w' belongs to $\widetilde{P}_{\text{PO}}$. Since $w' \geq w$, all inequalities (4) are still satisfied by (w', x). Also, inequalities (5) and (6) must be satisfied, because $w' \in \widetilde{P}_{\text{PO}}$. Thus, (w', x) is a feasible point of $\widetilde{P}_{\text{GOM–SD}}$.

Conversely, suppose that $x \notin \widetilde{P}_{\text{AM–SD}}$. We assume that $0 \leq x \leq 1$, otherwise the proof is trivial. This implies that at least one of the inequalities (2) is violated. Let C be a cycle in G_O such that the associated inequality (2) is not satisfied. In order to satisfy inequalities (4), for each arc $x_r \in R_C$, we must set $w_{d_r j} \geq x_r$ for every disk $j \in S_r$. But this means that w must violate at least one of the inequalities (7). Therefore, there exists no $w \in \widetilde{P}_{\text{AC}}$ such that (w, x) satisfies inequalities (4). We then apply Proposition 2 and conclude that $\widetilde{P}_{\text{AM–SD}} = \text{Proj}_x(\widetilde{P}_{\text{GOM–SD}})$.

Now, we note that $\text{Proj}_x(P_{\text{GOM–SD}}) = \widetilde{P}_{\text{AM–SD}} \cap \mathbb{Z}^{|R|}$ and the equality $P_{\text{AM–SD}} = \text{Proj}_x(P_{\text{GOM–SD}})$ follows. □

We now use the following results due to Balas and Oosten [11]. Consider the polytope $Q := \{(w, x) \in \mathbb{R}^p \times \mathbb{R}^q : Aw + Bx \leq b\}$. We partition the rows of (A, B, b) into $(A^=, B^=, b^=)$ and $(A^\leq, B^\leq, b^\leq)$, where the former represents the equality subsystem of Q. Let $F := \{(w, x) \in Q : \alpha w + \beta x = \beta_0\}$ be a facet of Q. Also, define $r^* := \text{rank}(A^=)$ and $r^*_F = \text{rank}\binom{\alpha}{A^=}$.

Theorem 4 (*Balas and Oosten [11]*)**.** $\dim(\text{Proj}_x(Q)) = \dim(Q) - p + r^*$.

Theorem 5 (*Balas and Oosten [11]*)**.** *Given a facet F of Q, $\text{Proj}_x(F)$ is a facet of $\text{Proj}_x(Q)$ if and only if $r^*_F = r^*$.*

Because the GOM models do not have an equality subsystem, $A^=$ is vacuous, so we have $r^* = 0$ and $r^*_F = \text{rank}(\alpha)$. Thus, a facet of $\widetilde{P}_{\text{GOM–SD}}$ projects into a facet of $\widetilde{P}_{\text{AM–SD}}$ if and only if $\alpha = 0$.

Proposition 6. *The dimension of $P_{\text{AM–SD}}$ is $|R|$.*

Proof. As showed by Kunigami et al. [4], the dimension of the polytopes defined by the GOM models is $|S|(|S| - 1) + |R|$. We then use Theorem 4 and the result follows. □

The inequalities in the next propositions have all been proved to be facet-defining for the GOM models by Kunigami et al. [4]. The results follow directly from this fact together with Theorem 5.

Proposition 7. *Given an arc $r \in R$, the inequality $x_r \geq 0$ always defines a facet of $P_{\text{AM–SD}}$. The inequality $x_r \leq 1$ defines a facet if and only if $S_r = \emptyset$.* □

Consider a pair of arcs $r, s \in R$ that form a 2-cycle in G_O (i.e., such that $d_r \in S_s$ and $d_s \in S_r$). For both SDP and PRDP, r and s cannot be visible simultaneously in a solution because either d_r covers s or d_s covers r. Let G_R be an undirected graph with a vertex v_r for each arc $r \in R$ and an edge $\{v_r, v_s\}$ for each pair of arcs r and s that form a 2-cycle in G_O.

Proposition 8. *Given a maximal clique K in G_R, the inequality*

$$\sum_{r\,:\,v_r \in K} x_r \leq 1$$

defines a facet of $P_{\text{AM–SD}}$.

To conclude this section, it is not clear if the proof of Theorem 3 can be adapted for PRDP. To do that, we would have to guarantee that all cycles used in Proposition 1 are in $\widehat{\mathcal{C}}$, which does not seem trivial. However, since $P_{\text{AM–SD}}$ is full-dimensional and $P_{\text{AM–SD}} \subseteq P_{\text{AM–PRD}}$, $P_{\text{AM–PRD}}$ is also full-dimensional. Moreover, the inequalities in Propositions 7 and 8 are also valid for PRDP and, therefore, define facets of $P_{\text{AM–PRD}}$.

3.2. Facet-defining cycle inequalities

Inequalities (2) and (3) are generally not facet-defining and lifting procedures to strengthen them are discussed in Section 6. In this section, we present necessary and sufficient conditions under which these inequalities are, indeed, facet-defining. The result is valid for both types of drawings.

Theorem 9. *Consider a cycle $C \in \mathcal{C}$ (resp. $C \in \widehat{\mathcal{C}}$). The following conditions are necessary and sufficient for the cycle inequality associated to C to define a facet of $P_{\text{AM–SD}}$ (resp. $P_{\text{AM–PRD}}$):*

1. *The subgraph $G_O[R_C]$ contains a single cycle from $\mathcal{C}$ (resp. $\widehat{\mathcal{C}}$);*

2. *For each arc $s' \notin R_C$ there exists an arc $s \in R_C$ such that the subgraph $G_O[R_C - \{s\} + \{s'\}]$ has no cycles from $\mathcal{C}$ (resp. $\widehat{\mathcal{C}}$).*

Proof. *Necessity*: Suppose condition 1 is not satisfied (see Fig. 9 for an example). Let $C' \neq C$ be another cycle from $\mathcal{C}$ (resp. $\widehat{\mathcal{C}}$) contained in $G_O[R_C]$. Clearly, $R_{C'} \subseteq R_C$ and inequality (16) is valid.

$$\sum_{r \in R_{C'}} x_r \leq |C'| - 1. \tag{16}$$

Now, for each $r \in R_C - R_{C'}$, we add inequality $x_r \leq 1$ to (16) to obtain

$$\sum_{r \in R_{C'}} x_r + \sum_{r \in R_C - R_{C'}} x_r \leq |C'| - 1 + |C| - |C'|$$

which can be rewritten as

$$\sum_{r \in R_C} x_r \leq |C| - 1.$$

The last inequality is precisely the one associated with cycle C. Since it was obtained by adding other valid inequalities, it cannot be facet-defining.

Suppose now that condition 2 is not satisfied for some arc $s' \notin R_C$ (see Fig. 10 for an example). This means that for every arc $s \in R_C$, $G_O[R_C - \{s\} + \{s'\}]$ contains a cycle from $\mathcal{C}$ (resp. $\widehat{\mathcal{C}}$). Note that to satisfy the cycle inequality associated with C at equality, $C-1$ arcs from R_C must be visible simultaneously. Thus, we conclude that equality will never be achieved if s' is visible, which implies that the following inequality is valid:

$$x_{s'} + \sum_{r \in R_C} x_r \leq |C| - 1.$$

The latter inequality is stronger than the original cycle inequality which, therefore, cannot be facet-defining.

Sufficiency: Suppose that both conditions are satisfied. We will show that inequality (17) defines a facet of $P_{\text{AM–SD}}$ (resp. $P_{\text{AM–PRD}}$).

$$\sum_{r \in R_C} x_r \leq |C| - 1. \tag{17}$$

Consider inequality (18) and assume that it is valid for $P_{\text{AM–SD}}$ (resp. $P_{\text{AM–PRD}}$).

$$\sum_{r \in R} \alpha_r x_r \leq \alpha_0. \tag{18}$$

Let $\mathcal{F}_C$ and $\mathcal{F}_\alpha$ be the faces of $P_{\text{AM–SD}}$ (resp. $P_{\text{AM–PRD}}$) induced by (17) and (18), respectively. Suppose that $\mathcal{F}_C \subseteq \mathcal{F}_\alpha$. We will show that (18) is a multiple of (17) and, therefore, (17) is facet-defining.

Note that, by condition 1, any subset of $|C| - 1$ arcs from R_C can be visible simultaneously in a solution. Thus, for each arc $s \in R_C$ we may create a valid solution that belongs to $\mathcal{F}_\alpha$ by assigning $x_r = 1$ for all $r \in R_C - \{s\}$ and $x_r = 0$ to all other variables. Each of these solutions can be replaced in (18) and we obtain

$$\sum_{r \in R_C - \{s\}} \alpha_r = \alpha_0. \tag{19}$$

Adding all Eqs. (19) for each $s \in R_C$ yields:

$$\sum_{r \in R_C} \alpha_r = \frac{|C|}{(|C| - 1)} \alpha_0. \tag{20}$$

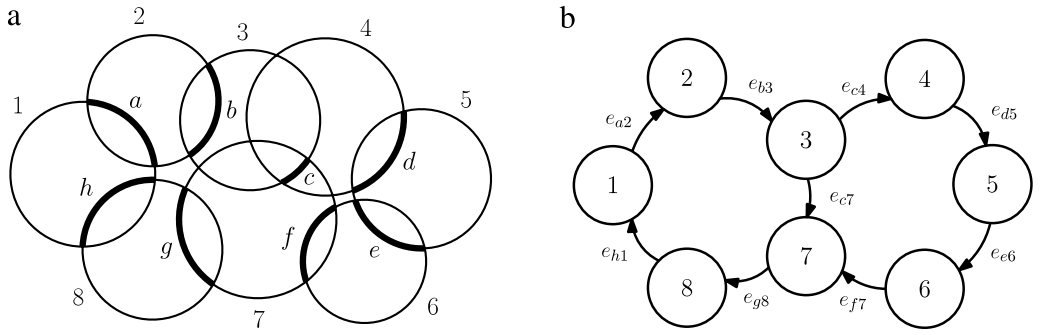

Fig. 9. (a) An arrangement with a cycle $C \in \mathcal{C}$ induced by arcs a–h; (b) the corresponding graph $G_O[R_C]$ that satisfies condition 2 but not 1 for stacking drawings.

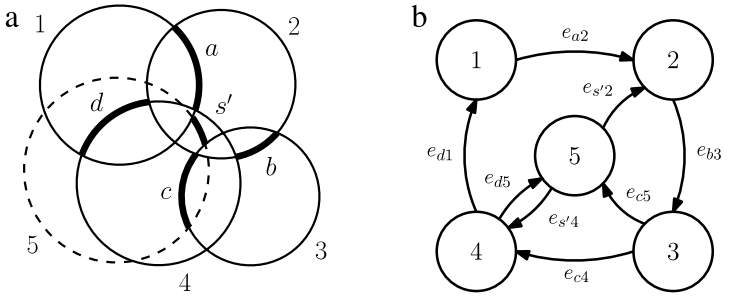

Fig. 10. (a) An arrangement with a cycle C induced by arcs a–d and another arc $s' \notin R_C$; (b) the corresponding graph $G_O[R_C + \{s'\}]$. Although $G_O[R_C]$ satisfies condition 1, there exists no $s \in R_C$ such that $G_O[R_C - \{s\} + \{s'\}]$ satisfies condition 2 for either types of drawings.

Now, for each $s \in R_C$, we subtract the corresponding Eq. (19) from (20) to obtain

$$\alpha_s = \frac{\alpha_0}{(|C| - 1)},$$

which is the desired value for the coefficients.

We are left to show that for every other $s' \notin R_C$, $\alpha_{s'} = 0$. By condition 2, there exists an arc $s \in R_C$ such that the subgraph $G_O[R_C - \{s\} + \{s'\}]$ has no cycles from $\mathcal{C}$ (resp. $\widehat{\mathcal{C}}$). A valid solution that belongs to $\mathcal{F}_\alpha$ can be created by setting $x_r = 1$ for all $r \in R_C - \{s\} + \{s'\}$ and assigning zero to all other variables. We replace this new solution in (18) and obtain

$$\sum_{r \in R_C - \{s\} + \{s'\}} \alpha_r = \alpha_0. \tag{21}$$

We can finally combine (21) with Eq. (19) and the result follows. □

4. Initial set of inequalities

Since there might be an exponential number of inequalities (2) and (3), it is necessary to select a subset of them to form the initial model that will be loaded into the ILP solver. In our implementation, we use those associated with 2-cycles, which only comprise a polynomial number of inequalities and are valid for both types of drawings. However, graph G_O is often quite dense and the number of 2-cycles is quadratic in $|R|$, yielding $O(n^4)$ constraints. Thus, we cannot include them all because the model becomes too big. Note that the simple alternative of starting with an empty model has the same effect: all 2-cycle inequalities

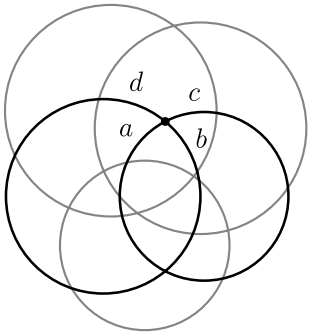

Fig. 11. A non-degenerate vertex and four arcs incident to it.

will be violated and added by our separation routines. We next describe a way to select a subset of these constraints that still provide good dual bounds.

Our strategy is based on the following inequalities studied by Kunigami et al. [5]. Consider a non-degenerate vertex of the arrangement (i.e., one that is an intersection point of exactly two disks), as shown in Fig. 11. Note that every disk that contains arc c must also contain arc a. Therefore, if a is visible in a drawing, so is c. By symmetry, the same observation holds for arcs b and d. This allows us to write the inequalities (22). We remark that both polyhedra $P_{\text{AM–SD}}$ and $P_{\text{AM–PRD}}$ are monotone and, consequently, these inequalities are not valid in general. However, they must hold for any optimal solution and can be used to speed up the resolution of the models.

$$\begin{aligned} x_a &\leq x_c \\ x_b &\leq x_d. \end{aligned} \tag{22}$$

We now examine the two disks i and j shown in Fig. 12 and the named arcs on their boundaries. Note that any pair formed by a named arc from i and another from j corresponds to a 2-cycle in G_O, potentially leading to 49 inequalities. Nevertheless, only the ones associated with arcs r and s need to be included in the model. Using constraints (22), we obtain:

$$\begin{aligned} x_r &\geq x_{r_1} \geq x_{r_3} \geq x_{r_5}, \\ x_r &\geq x_{r_2} \geq x_{r_4} \geq x_{r_6}, \\ x_s &\geq x_{s_1} \geq x_{s_3} \geq x_{s_5}, \\ x_s &\geq x_{s_2} \geq x_{s_4} \geq x_{s_6}. \end{aligned}$$

Thus, including the inequality $x_r + x_s \leq 1$ together with (22) we guarantee that all 48 other constraints will also be satisfied. We say that arcs r and s *dominate* arcs r_k and s_k ($k = 1, 2, \ldots, 6$), respectively, because in any feasible solution we must have $x_r \geq x_{r_k}$ and $x_s \geq x_{s_k}$. We refer to arcs which are not dominated by any other (such as r and s) as *maximal arcs*.

Each 2-cycle in G_O corresponds to a 2-clique in G_R (defined in Section 3). Therefore, using Proposition 8, we conclude that an inequality associated with one of these cycles defines a facet if and only if the corresponding clique K in graph G_R is maximal. To take advantage of this fact, before inserting an inequality into the solver, we run a fast greedy procedure that iteratively increases K. This is done in two steps: initially, only maximal arcs are inserted in K; after this is no longer possible, all other arcs are allowed. The algorithm works by computing, at each iteration, the set V_C of candidate vertices (i.e., those that are adjacent to every member of K). Then, it chooses the candidate that has the largest number of neighbors in V_C, updates K, and continues until a maximal clique is obtained.

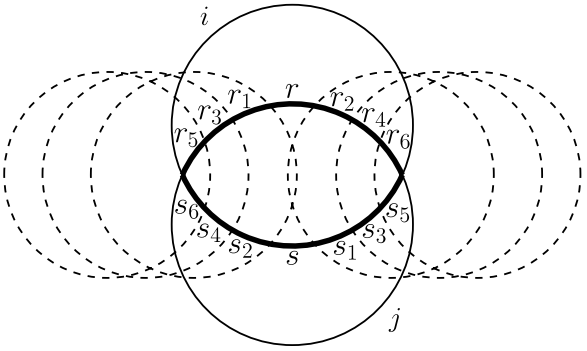

Fig. 12. Two disks i and j and seven arcs on the boundary of each one of them, which give rise to 49 2-cycles in G_O.

5. Separation routines

Let x^* be the optimal solution to the linear relaxation of the arc model at some node in the branch-and-bound tree. For each type of drawing, the objective of the separation routine is to find a cycle $C \in \mathcal{C}$ or $C \in \widehat{\mathcal{C}}$ such that the corresponding cycle inequality (2) or (3) is violated, i.e., such that

$$\sum_{r \in R_C} x^*_r > |C| - 1. \tag{23}$$

Before we proceed, let us associate with each edge $e_{rj} \in E(G_O)$ a weight $w(e_{rj}) = 1 - x^*_r$. Also, to simplify the notation, given a subgraph H_O of G_O, we define $w(H_O) := \sum_{e_{rj} \in H_O} w(e_{rj})$. We can rewrite (23) as (24).

$$w(C) < 1. \tag{24}$$

For SDP, all cycles in G_O lead to valid inequalities, so we simply need to find one that satisfies (24). The algorithm we use was first presented by Grötschel et al. [9] for the acyclic subgraph polytope. We can determine if such a cycle exists by using a shortest path algorithm. For each pair of vertices $v_i, v_j \in V(G_O)$, we compute a shortest path from v_i to v_j and another from v_j to v_i. If the sum of the weights of the paths is less than one, we join them to obtain the required cycle. If no such pair of vertices exist, all cycle inequalities are satisfied. Thus, (2) can be separated in polynomial time.

For PRDP, care must be taken to ensure that only cycles from $\widehat{\mathcal{C}}$ are prohibited. We do this with a weighted version of the algorithm that computes face-transitive closures. Recall that in the original procedure, given a subgraph H_O of G_O, we repeatedly compute the transitive closure of $H_O[S_f]$ for each maximal face $f \in F$. This is equivalent to the following: given two vertices $v_i, v_j \in V(H_O)$, if there is a path π_{ij} from v_i to v_j in a subgraph $H_O[S_f]$, we must add an edge (v_i, v_j). For the separation routine, we do exactly the same, with the additional detail that the newly created edge is assigned a weight $w(i, j) := w(\pi_{ij})$. If the edge already exists, we need only update its weight, choosing the smallest value between the current $w(i, j)$ and $w(\pi_{ij})$. This procedure is repeated until there are no more changes in the weights of the edges. It is not clear how many iterations might be necessary until this happens, so the computational complexity of this algorithm is open. Nevertheless, our experiments showed that it performs very well in practice.

We can determine that a cycle C is not physically realizable if some subgraph $C[S_f]$ has a cycle. Note that this implies that the transitive closure of $C[S_f]$ must also contain a 2-cycle, because the transitive closure of a cycle is a complete graph and at least a pair of reverse edges will occur. Similarly, in the separation routine, we can identify a violated inequality if there is a 2-cycle with weight less than 1. To recover the original set of arcs that gave rise to the cycle, we store, with each new edge (v_i, v_j), the set of arcs in the path π_{ij} that led us to create it. Naturally, this set must be updated whenever the weight of an edge changes.

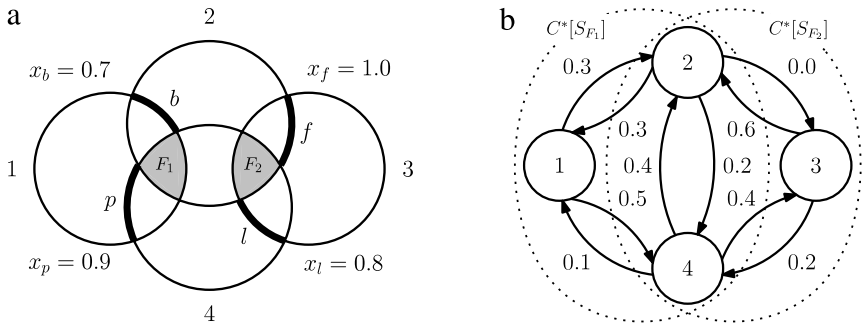

Fig. 13. (a) An arrangement with four arcs b, f, l, and p, and the values of the associated variables; (b) the weighted graph obtained after running the separation routine for PRDP.

The procedure is illustrated in Fig. 13. The arrangement of four disks is the same as in Fig. 5(a). For the sake of clarity, this example only takes into account the arcs that give rise to the new inequality. Nevertheless, the actual algorithm must be applied to all arcs of the arrangement. The graph on the right shows the weights obtained after running the separation routine. We may take any 2-cycle to create a valid constraint, so for this example we choose the one between disks 2 and 4. Its weight is 0.6, so the associated inequality is indeed violated. Arcs f and l induce the path 2–3–4, which leads us to create edge $(2, 4)$. Likewise, edge $(4, 2)$ was created due to arcs p and b. This yields the inequality $x_b + x_f + x_l + x_p \leq 3$.

During the execution of the branch and cut algorithm, we often come across several inequalities that are satisfied with a very small slack. When this is the case, it is possible that these inequalities become violated after the execution of lifting procedures. To take advantage of this fact, we replace the right hand side of (23) by $|C| - 2$ and modify the separation routines so that they look for cycles of weight $w(C) < 2$. All inequalities found are subjected to lifting procedures (described in Section 6) and are added to the ILP solver only if they are violated. This strategy led to a significant improvement in both dual bounds and execution times.

6. Lifting

In this section, we present efficient methods to strengthen inequalities (2) and (3). The discussion that follows is applicable for both types of drawings. Consider a cycle C in G_O formed by a set of arcs R_C (as defined in Section 2.1). We assume that $C \in \mathcal{C}$ or $C \in \widehat{\mathcal{C}}$ depending on the problem variant being addressed, so the cycle inequality $\sum_{r \in R_C} x_r \leq |C| - 1$ is valid. Let $R_l := R - R_C$ be the set of candidates for lifting. A lifting procedure must compute, for each arc $r \in R_l$, a non-negative integral coefficient α_r, such that inequality (25) is satisfied by all feasible solutions x.

$$\sum_{r \in R_C} x_r + \sum_{r \in R_l} \alpha_r x_r \leq |C| - 1. \tag{25}$$

Lifting procedures usually have two main concerns. Firstly, one wishes to make the coefficients as large as possible, so the resulting inequalities become stronger. However, computing the absolute best values is often a difficult problem. Secondly, there may be many different sets of coefficients that lead to strong inequalities. To deal with the first issue, we use two fast heuristics that guarantee the validity of the lifted inequalities but not the optimality of the coefficients. As for the second issue, we opt to speed up the process by creating a single constraint for each cycle C. Our heuristics rely mainly on identifying pairs of arcs $r, s \in R$ that form 2-cycles in G_O. Setting any of them visible automatically forces the other one to be covered, so we say that r *blocks* s, and vice versa.

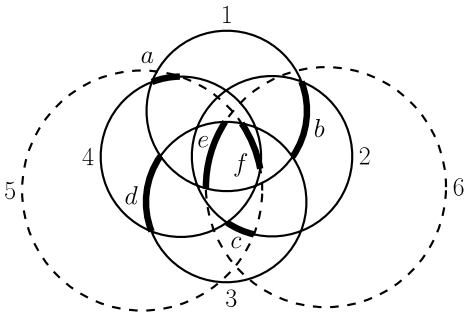

Fig. 14. A cycle formed by arcs a, b, c and d, and two other arcs e and f whose coefficients can be lifted with the first heuristic.

The first heuristic is based on the following observation: if two arcs cannot be visible simultaneously, their coefficients can be determined independently. Therefore, we can compute each α_r as if x_r were the only lifted variable. More precisely, we want to assign a value to α_r such that

$$\alpha_r x_r + \sum_{r' \in R_C} x_{r'} \leq |C| - 1 \tag{26}$$

is satisfied by every feasible solution x. This clearly holds if $x_r = 0$, so suppose $x_r = 1$. There must be at least $\alpha_r + 1$ arcs from R_C covered in x. Let B_C^r be the set of arcs in R_C that are blocked by r. We can guarantee that inequality (26) is valid by setting $\alpha_r := |B_C^r| - 1$.

After this is done for all arcs in R_l, we need to choose a subset of arcs such that at most one of them is visible in any solution. Recall that graph G_R (defined in Section 3) has a vertex for each arc of the arrangement and its edges indicate pairs that block each other. It follows that a clique in G_R leads to a set of arcs with unrelated lifting coefficients. We then apply the greedy algorithm from Section 4 to find a clique K in G_R, taking care to include a vertex v_r only if $\alpha_r > 0$. Finally, we are able to generate the lifted inequality $\sum_{r \in R_C} x_r + \sum_{v_r \in K} \alpha_r x_r \leq |C| - 1$.

The procedure is illustrated in Fig. 14. Arcs a through d induce a cycle between disks 1–4, which are shown with solid boundary. The dashed disks 5 and 6 introduce two arcs e and f that block each other and, thus, have unrelated lifting coefficients. We have $B_C^e = \{b, c\}$ and $B_C^f = \{a, c, d\}$, which yields the valid lifted inequality $x_a + x_b + x_c + x_d + x_e + 2x_f \leq 3$.

In the second heuristic, we try to lift the coefficients of arcs that do not necessarily block each other. Let us rewrite (25) as (27).

$$\sum_{r \in R_l} \alpha_r x_r \leq \sum_{r \in R_C} (1 - x_r) - 1. \tag{27}$$

We start by showing how to assign values to each α_r in such a way that the relaxed inequality

$$\sum_{r \in R_l} \alpha_r x_r \leq \sum_{r \in R_C} (1 - x_r) \tag{28}$$

is satisfied by every feasible solution. Once this is done, a valid constraint can be obtained by decrementing the values of (some) coefficients.

Two points should be noted about inequality (28). Firstly, the summation on its right-hand side is simply the number of arcs from R_C that are hidden in a given solution x. Secondly, each visible arc $r \in R_l$ contributes with α_r units to its left-hand side. Thus, to satisfy (28), each visible arc $r \in R_l$ must be compensated by at least α_r covered arcs from R_C.

The main idea behind our heuristic is to select, for each $r \in R_l$, a set of arcs from R_C that are blocked by r (and, therefore, hidden whenever r is visible). We refer to this set as the *compensating set* of r and

denote it by M_C^r. The value of each coefficient α_r is then set to $|M_C^r|$. If two lifting candidates can be visible simultaneously, their compensating sets must be disjoint; otherwise, they may share some common elements. Next, we show that if this condition is obeyed, our choice of coefficients guarantees that (28) holds for all feasible solutions.

Initially, observe that if an arc $r \in R_l$ is visible, at least the arcs in M_C^r must be covered. Hence, inequality (29) is always satisfied.

$$\left| \bigcup_{\substack{r \in R_l \\ x_r = 1}} M_C^r \right| \leq \sum_{r \in R_C} (1 - x_r)\,. \tag{29}$$

We now analyze the left-hand side of (29). Since we assume that the preceding condition is obeyed, all sets in the union are disjoint. This allows us to write

$$\left| \bigcup_{\substack{r \in R_l \\ x_r = 1}} M_C^r \right| = \sum_{\substack{r \in R_l \\ x_r = 1}} |M_C^r|.$$

But the last summation just adds the coefficients of visible arcs. Thus, it is equal to $\sum_{r \in R_l} \alpha_r x_r$ and the result follows.

So far we are able to satisfy only (28) but not (27). A straightforward solution would be to decrement all positive coefficients by one. However, it is possible to do better than this. If we can make sure that equality never occurs in (29), inequality (27) will be automatically satisfied. One way to do this is to prohibit one arc from each B_C^r from being included in any compensating set. This way, each time arc r is visible, there is at least one hidden arc that is not accounted for in the left-hand side of (29). More precisely, we want to select a set of arcs $H_C \subseteq R_C$ such that $H_C \cap B_C^r \neq \emptyset$ for all $r \in R_l$. This is simply the statement of a hitting set problem.

We are now able to describe our algorithm. We first select the arcs $r \in R_l$ such that $|B_C^r| \geq 2$. There is no need to consider the others because our heuristic cannot assign a positive coefficient to any of them. We then construct the set B_C^r for each selected arc r. A hitting set H_C is found using a greedy heuristic that selects, at each stage, the element that is contained in the largest number of sets that have not yet been hit. We make H_C minimal by excluding unnecessary elements.

Next, we start building the compensating sets. Given an arc $s \in R_C - H_C$, let B_l^s denote the set of arcs from R_l that block s. We need to choose an arc from B_l^s and insert s into its compensating set. We wish to create an inequality with a large violation, so our strategy is to select the arc r with largest x_r. Recall that we are allowed to insert s in more than one compensating set as long as the associated arcs cannot be visible together in a feasible solution. So, instead of picking a single arc from B_l^s, we select a set of arcs such that any two of them block each other. Again, such a set of arcs corresponds to a clique in graph G_R and we can compute it the same way we did in the first heuristic. Finally, we generate the lifted inequality $\sum_{r \in R_C} x_r + \sum_{r \in R_l} |M_C^r| x_r \leq |C| - 1$.

The procedure is depicted in Fig. 15. Arcs a through d induce a cycle between disks 1–4, which are shown with solid boundary. The dashed disks 5, 6 and 7 introduce three arcs e, f and g whose coefficients we want to lift. We have $B_C^e = \{a, b, c\}, B_C^f = \{c, d\}$ and $B_C^g = \{b, c\}$. Thus, a possible hitting set is $H_C = \{c\}$. For the remaining arcs, we have $B_l^a = \{e\}, B_l^b = \{e, g\}$ and $B_l^d = \{f\}$. We can now create the compensating sets. Arcs a and d can only be inserted in M_C^e and M_C^f, respectively. Arc b can be inserted in either M_C^e or M_C^g, but not both because e and g do not block each other. For this example, we choose to insert it into M_C^g. Finally, the compensating sets are $M_C^e = \{a\}, M_C^f = \{d\}$ and $M_C^g = \{b\}$, which results in the lifted inequality $x_a + x_b + x_c + x_d + x_e + x_f + x_g \leq 3$.

As we show in Section 7, these lifting heuristics contribute a great deal to improving the dual bounds generated by our formulations. Although no guarantees are given in general as for the dimension of the faces

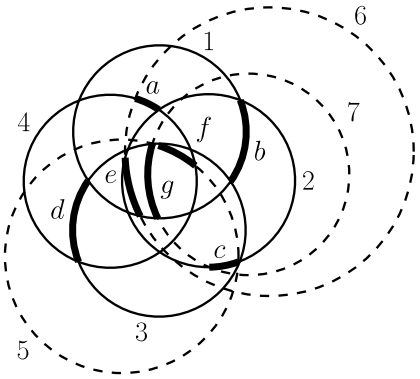

Fig. 15. A cycle formed by arcs a, b, c and d, and three other arcs e, f and g whose coefficients can be lifted with the second heuristic.

induced by the lifted inequalities, it can be easily shown that those obtained for Figs. 14 and 15 are, indeed, facet-defining (the proof is omitted for the sake of brevity).

7. Computational experiments

We assess the effectiveness of our solution approach through a series of computational experiments with 60 real-world instances. Firstly, we evaluate the performance of our strategy to select the initial set of inequalities. Secondly, we measure the impact of lifting on both the dual bounds and the total running times. Finally, we compare models AM and GOM for the three problem variants addressed in this paper. For the sake of brevity, the tables show the results for the most challenging instances. Our complete set of instances is available in our web page [12].

7.1. Implementation details

The procedures were implemented in C++ and compiled with GCC 4.4.3. We build the arrangements with CGAL 4.0 using exact arithmetic. The ILPs are solved with CPLEX 12.4. We apply a branch-and-cut algorithm and use the conventional search supplied by CPLEX. We limit the time spent by the solver to at most five hours per component. The experiments were run on an Intel Xeon X3430, 2.40 GHz CPU with 8 GB RAM.

All instances are decomposed in a preprocessing phase with the techniques from [4]. We solve all components and then recombine them to create an optimal solution for the whole instance. For the Max-Total variants, solution values and gaps are calculated disregarding arcs that are visible in every feasible drawing. Primal solutions for the Max-Total and the Max–Min problems are generated with the GRASP heuristic from [7] and with the greedy algorithm from [3], respectively.

As observed by Kunigami et al. [4], it is sometimes possible to bypass the resolution of integer programs when solving the Max–Min PRDP. If an instance (or a component thereof) has a face that is contained in every disk, all physically realizable solutions must also be stacking drawings. This allows us to use the polynomial time algorithm from Cabello et al. [3]. Besides that, not all components need to be solved optimally. We just need an optimal solution to the one that determines the Max–Min value. For the others, it often suffices to find a good heuristic solution, which can once again be done with the algorithm from [3].

7.2. Problem instances

We test our model on three sets of instances. The first set was presented by Cabello et al. [3] and contains four instances: *City 156* and *City 538*, which represent the 156 and 538 largest American cities,

Table 1
Results for the initial linear relaxation using the model with all 2-cycle inequalities and the reduced one based on maximal arcs. Times are given in seconds.

Instance	$\|S\|$	$\|R\|$	% Gaps		Number of inequalities			Relaxation times		
			All	Max.	All	Max.	Ratio	All	Max.	Ratio
City 538	538	7,516	0.26%	0.87%	870,660	54,037	16.1	9878	68	145.3
Magnitudes	602	13,289	0.13%	0.79%	597,763	47,520	12.6	98	4	24.5
Brazil-S1	150	2,568	0.00%	0.47%	147,597	9,975	14.8	54	1	54.0
France-S1	135	3,230	0.55%	2.57%	735,188	24,256	30.3	4226	17	248.6
Greece-S1	102	3,482	0.87%	4.85%	694,198	27,412	25.3	5341	19	281.1
Italy-S1	300	4,366	0.00%	0.35%	246,154	20,036	12.3	206	4	51.5
Japan-S1	150	3,544	0.00%	0.79%	269,272	18,101	14.9	231	4	57.8
Portugal-S1	150	5,070	0.01%	0.72%	636,866	36,380	17.5	1691	21	80.5
USA(West)-S1	87	3,717	0.00%	0.84%	559,128	27,880	20.1	2992	20	149.6
Brazil-S2	150	3,294	0.02%	0.20%	265,447	13,697	19.4	97	2	48.5
France-S2	135	3,354	0.77%	1.32%	769,642	21,661	35.5	>5 h	14	>1285.7
Germany-S2	150	3,299	0.00%	0.32%	241,062	21,506	11.2	276	10	27.6
Greece-S2	102	3,335	0.17%	3.67%	587,597	21,048	27.9	4140	12	345.0
Italy-S2	300	6,725	0.04%	0.30%	580,512	34,109	17.0	962	14	68.7
Japan-S2	150	4,954	0.00%	0.23%	592,618	36,592	16.2	3520	29	121.4
Poland-S2	300	4,736	0.01%	0.21%	293,886	25,358	11.6	372	8	46.5
Portugal-S2	150	6,656	0.00%	0.15%	1,306,159	55,218	23.7	9083	56	162.2
Spain-S2	300	6,497	0.06%	0.58%	297,607	25,093	11.9	82	3	27.3
Switzerland-S2	186	4,945	0.04%	0.61%	259,127	24,870	10.4	553	12	46.1
USA(West)-S2	87	4,465	0.00%	0.88%	871,332	33,588	25.9	4698	32	146.8

respectively; and *Deaths* and *Magnitudes*, which represent the death count and the Richter scale magnitude of 602 earthquakes worldwide, respectively.

The two other sets were used in [4,7] and contain 28 instances each. They were generated from data on the population of cities from several countries. We refer to them as *Population-S1* and *Population-S2*. They are both based on the same data, but with different proportionality constants between the area of the disks and the numerical values they represent. The instances in Population-S2 are scaled in such a way that the area of each disk is twice as large as that in the analogous instance from Population-S1.

7.3. Numerical results

We first address the issue of selecting the initial set of inequalities that are loaded into the ILP solver. We compare the technique based on maximal arcs with the alternative of using all 2-cycle inequalities. We remark that, as mentioned in Section 4, the latter is similar to starting with an empty model. It should also be noted that the techniques are the same for both stacking and physically realizable drawings. For this reason, it is not necessary to show the results for each problem variant. In the text that follows, duality gaps are calculated with respect to Max-Total SDP optimal values.

Table 1 shows the results obtained after solving the linear relaxation of the initial set of inequalities. For each instance, we report the number of disks and arcs, the duality gap, the number of inequalities in the linear program and the time spent solving the relaxation. Columns *All* and *Max* refer to the model with all 2-cycle inequalities and to the reduced one based on maximal arcs, respectively. We also present the ratios between the results of the two alternatives.

Using the reduced set of constraints increased the duality gaps by only 0.60% on average. The largest increase occurred for instance Greece-S1 and, still, it was less than 4.00%. On average, the reduced model has 8.5 times fewer inequalities and the relaxation is solved 23.2 times faster. The most remarkable reduction occurs for instance France-S2, which runs more than 1285 times faster with an increase of only 0.55% in the duality gap. Due to the effectiveness of this strategy, all experiments that follow use this reduced set of inequalities.

In the next experiment, we evaluate the impact of lifting on the dual bounds produced by our model. For each instance, we solve the root node of the branch-and-bound tree, which involves solving a linear

Table 2
Duality gaps after solving the root node of the branch-and-bound tree with and without lifting.

Instance	$\|S\|$	$\|R\|$	Max-Total SDP		Max-Total PRDP		Max–Min PRDP	
			GOM/AM	AM++	AM	AM++	AM	AM++
City 538	538	7,516	0.57%	0.02%	0.56%	0.02%	60.18%	59.20%
Magnitudes	602	13,289	0.59%	0.04%	0.59%	0.05%	109.42%	107.17%
Egypt-S1	98	2,033	0.95%	0.04%	0.95%	0.11%	62.51%	60.95%
France-S1	135	3,230	2.40%	0.00%	2.40%	0.00%	83.77%	75.45%
Greece-S1	102	3,482	3.81%	0.00%	3.81%	0.00%	81.91%	74.21%
Italy-S1	300	4,366	0.23%	0.01%	0.23%	0.01%	58.15%	55.72%
Japan-S1	150	3,544	0.24%	0.00%	0.24%	0.00%	71.10%	68.44%
Portugal-S1	150	5,070	0.60%	0.00%	0.60%	0.00%	49.22%	46.90%
USA(West)-S1	87	3,717	0.38%	0.00%	0.38%	0.00%	50.63%	47.30%
Canada-S2	150	2,790	0.57%	0.05%	0.57%	0.05%	35.36%	34.57%
France-S2	135	3,354	1.08%	0.01%	1.08%	0.01%	97.60%	87.39%
Greece-S2	102	3,335	2.99%	0.00%	2.99%	0.00%	86.20%	80.59%
Israel-S2	150	4,222	1.03%	0.26%	1.03%	0.25%	44.48%	44.48%
Italy-S2	300	6,725	0.15%	0.06%	0.21%	0.12%	75.57%	70.82%
Poland-S2	300	4,736	0.11%	0.02%	0.11%	0.01%	45.75%	14.59%
Russia-S2	150	3,145	0.36%	0.02%	0.36%	0.02%	39.49%	0.00%
Switzerland-S2	186	4,945	0.45%	0.00%	0.45%	0.00%	67.54%	64.52%

relaxation and then executing our separation routines. This procedure is repeated until no more violated inequalities are found. Automatic cut generation routines were disabled during this experiment. The results are shown in Table 2. The original arc model is denoted by AM, whereas the improved formulation obtained after the inclusion of lifting is denoted by AM++. We report the duality gaps after solving the root node of the enumeration tree.

The results show that the behavior for both Max-Total problems is very similar. Although AM leaves a small gap, it can still lead to a relatively costly branching. The addition of lifting makes all gaps drop to less than 0.26% and 18 instances that would otherwise need branching can be optimally solved at the root node. As for the Max–Min PRDP, due to the need of a continuous variable to model the objective function, duality gaps are significantly larger. Nevertheless, the lifting procedures are still able to reduce the gaps for many instances.

In the final experiment, we run models GOM, AM and AM++ on all instances for the three problem variants. We enable automatic cut generation routines to evaluate the performance of the models in conjunction with the most advanced features provided by the solver. The results are shown in Tables 3–5. Optimal values are given in column *Opt*. For each model, we report the total running time (in seconds), the number of cuts N_C added by our separation routines and the number of nodes N_N solved in the branch-and-bound tree. Some preliminary tests indicated that the GOM model also benefits from the inclusion of the initial set of inequalities described in this work. To establish a fair comparison between the models, the results that follow incorporate these improvements.

To further analyze the speedup enabled by each model, Table 6 shows, for each pair of models, the average, maximum and minimum ratio of their running times. To focus on the hardest instances, the results disregard those that can be solved in less than 10 s by both models. Average values are computed as the geometric mean of the running time ratios.

For both Max-Total variants, the results show that AM performs better than GOM for all instances. The reduction in the number of variables compensates for the exponential number of inequalities and the arc-based formulation achieves a (geometric) average speedup of 5.9 and 6.0 for the SDP and PRDP variants, respectively. The addition of lifting allows AM++ to run 1.9 times faster than AM for both types of drawings. Besides, all but two instances are solved in the root node of the branch-and-bound tree. The greatest improvements occurred for instances France-S1 and Greece-S1, for which AM++ is over 82 times faster than GOM for SDP.

It is interesting to note that, as shown in Table 2, AM is never able to completely solve the problem on the root node of the search tree. However, in this experiment, this was possible due to the inclusion of

Table 3
Comparison of models GOM, AM and AM++ for the Max-Total SDP. Times are given in seconds.

Instance	$\|S\|$	$\|R\|$	Opt.	GOM		AM			AM++		
				Time	N_N	Time	N_C	N_N	Time	N_C	N_N
City 538	538	7 516	503.27	15 725	612	3851	4194	336	332	2753	1
Magnitudes	602	13 289	12 174.09	315	126	42	298	25	22	991	1
France-S1	135	3 230	964.07	5 039	276	435	596	50	61	1531	1
Greece-S1	102	3 482	1102.55	5 755	244	495	885	68	70	2361	1
Japan-S1	150	3 544	1911.18	249	20	17	479	1	12	896	1
Portugal-S1	150	5 070	1707.01	304	12	39	122	12	29	425	1
USA(West)-S1	87	3 717	1290.14	709	34	124	416	4	37	694	1
Denmark-S2	310	3 942	4289.57	120	32	30	1033	10	13	1433	1
Egypt-S2	98	3 043	2316.98	208	10	39	312	3	17	834	1
France-S2	135	3 354	1821.45	1 618	50	177	311	16	39	1177	1
Greece-S2	102	3 335	1806.11	3 024	285	418	891	195	43	1927	1
Israel-S2	150	4 222	2935.95	299	91	30	420	4	16	1276	1
Italy-S2	300	6 725	4775.34	262	8	53	433	18	37	1041	1
Japan-S2	150	4 954	3096.87	362	1	52	223	1	50	839	1
Norway-S2	150	2 881	2090.14	144	49	36	1828	43	15	2840	1
Poland-S2	300	4 736	9045.2	157	1	27	968	1	22	4518	1
Portugal-S2	150	6 656	2727.82	577	1	72	56	1	70	321	1
Switzerland-S2	186	4 945	5818.56	545	28	48	141	1	27	514	1
USA(East)-S2	150	4 524	3508.58	119	20	6	206	1	6	584	1
USA(West)-S2	87	4 465	1953.22	506	1	67	847	1	57	992	1

Table 4
Comparison of models GOM, AM and AM++ for the Max-Total PRDP. Times are given in seconds.

Instance	$\|S\|$	$\|R\|$	Opt.	GOM		AM			AM++		
				Time	N_N	Time	N_C	N_N	Time	N_C	N_N
City 538	538	7 516	503.27	15 378	538	2326	4882	377	260	3984	1
Magnitudes	602	13 289	12 174.09	439	315	42	654	12	24	1482	1
France-S1	135	3 230	964.07	5 034	276	369	954	50	63	2722	1
Greece-S1	102	3 482	1102.55	5 475	290	546	1488	100	70	3028	1
Japan-S1	150	3 544	1911.18	219	10	17	642	1	12	818	1
Portugal-S1	150	5 070	1707.01	306	24	40	224	28	29	544	1
USA(West)-S1	87	3 717	1290.14	709	34	123	556	4	37	804	1
Canada-S2	150	2 790	2922.57	108	37	9	360	8	6	756	1
France-S2	135	3 354	1821.45	1 366	22	142	436	14	39	1858	1
Germany-S2	150	3 299	2894.92	151	4	18	258	1	18	616	1
Greece-S2	102	3 335	1806.11	2 877	285	221	688	138	43	2678	1
Israel-S2	150	4 222	2935.95	402	94	32	440	8	16	988	1
Italy-S2	300	6 725	4775.34	259	16	40	444	1	34	986	1
Japan-S2	150	4 954	3096.87	340	1	53	340	1	50	1118	1
Norway-S2	150	2 881	2090.16	153	44	21	382	8	9	804	1
Poland-S2	300	4 736	9045.2	117	1	23	314	1	21	476	1
Portugal-S2	150	6 656	2727.82	542	1	72	100	1	70	366	1
Switzerland-S2	186	4 945	5818.56	630	32	61	220	1	26	640	1
USA(East)-S2	150	4 524	3508.58	97	10	7	226	1	7	534	1
USA(West)-S2	87	4 465	1953.22	512	1	63	660	1	58	1208	1

automatic cut generation routines. Nevertheless, it can be seen in Tables 3 and 4 that even when the number of nodes is one for both AM and AM++, the latter is still faster. This happens because our specialized lifting procedures are more effective than the generic cuts added by the solver.

As for the Max–Min PRDP, the arc models also perform better, though not as much as in the other variants. Models AM and AM++ achieve faster running times than GOM in 48 and 47 instances out of 60, respectively. On average, AM is 1.99 times faster than GOM and AM++ is 1.04 times faster than AM. This shows that using lifted inequalities is not always worthwhile. The reason is that the dual bounds for this version of the problem are a lot weaker than the ones for the Max-Total variants. As a consequence, the algorithm is highly dependent on branching and many linear relaxations must be solved. Since lifting increases the number of non-zeros in the ILP matrix, it might be better to use the weaker but sparser AM model.

Table 5
Comparison of models GOM, AM and AM++ for the Max–Min PRDP. Times are given in seconds.

Instance	$\|S\|$	$\|R\|$	Opt.	GOM		AM			AM++		
				Time	N_N	Time	N_C	N_N	Time	N_C	N_N
City 538	538	7516	0.76	6929	449	1607	1066	1 437	1469	1148	1 028
Egypt-S1	98	2033	9.44	184	705	971	180	18 584	923	460	7 854
France-S1	135	3230	6.45	>5 h	13 488	3319	400	18 120	1507	2932	8 382
Greece-S1	102	3482	8.88	>5 h	466	15 722	478	54 531	3754	3092	12 490
Italy-S1	300	4366	6.18	1890	1 021	1662	154	9 120	450	956	1 782
Japan-S1	150	3544	8.73	2216	1 091	403	518	799	672	958	2 873
Switzerland-S1	186	3047	15.94	1170	1 901	242	80	1 944	1223	352	10 809
USA(West)-S1	87	3717	9.31	>5 h	715	678	360	756	3463	1262	2 498
Brazil-S2	150	3294	15.82	473	44	35	256	18	45	590	48
Canada-S2	150	2790	10.68	131	27	11	256	1	11	374	1
Egypt-S2	98	3043	13.92	943	458	2099	588	3 155	1788	676	3 710
France-S2	135	3354	8.34	>5 h	3 897	7286	682	12 208	912	3766	1 107
Germany-S2	150	3299	12.36	5144	2 900	13 088	878	49 253	>5 h	1746	56 016
Greece-S2	102	3335	11.75	4314	11 942	970	556	2 735	>5 h	2116	144 981
Indonesia-S2	150	3478	8.43	663	1 284	2498	422	21 540	1320	1050	8 083
Israel-S2	150	4222	11.16	883	11 578	66	206	804	92	224	2 059
Italy-S2	300	6725	7.33	16 941	2 485	5956	712	3 918	2552	1656	1 216
Japan-S2	150	4954	13.64	522	1	103	188	1	153	762	1
Switzerland-S2	186	4945	17.78	>5 h	9 123	>5 h	754	15,859	>5 h	2166	11 743
USA(West)-S2	87	4465	12.05	13 205	5 057	17 916	532	39 147	>5 h	1674	8 803

Table 6
Summary of the results for all models and problem variants.

Problem	GOM/AM			AM/AM++			GOM/AM++		
	Avg.	Max.	Min.	Avg.	Max.	Min.	Avg.	Max.	Min.
Max-Total SDP	5.85	19.58	1.56	1.92	11.61	0.45	9.21	82.23	1.22
Max-Total PRDP	5.95	14.95	1.28	1.89	8.95	0.66	9.40	80.38	1.14
Max–Min PRDP	1.99	26.54	0.19	1.04	7.99	0.05	2.09	19.74	0.20

A single instance (Switzerland-S2) could not be solved by any of the models within the imposed time limit. The optimal value shown in Table 5 was verified after almost 54 h of computation with the AM model, which is the one that presented the best prospects after the initial five hours.

For the three problem variants, the number of cuts added by our separation routines range from a couple of hundreds to a few thousands. A comparison with the initial number of inequalities reported in Table 1 indicates that the size of the models is not significantly affected by the inclusion of these constraints. For most instances, AM++ adds considerably more cuts than AM. This behavior is expected because, as explained in Section 5, our lifting heuristics are applied not only to violated inequalities (2) and (3), but also to those that are satisfied with a small slack.

8. Conclusion

We study three variants of a proportional symbol maps problem and propose a novel ILP model in terms of arc variables only. We show that our model for stacking drawings is a projection of another one previously described in the literature. In addition, we describe a strategy to select the initial set of inequalities that greatly improves computation times not only with our models but also with another one from the literature. We also describe fast separation routines and effective lifting procedures. When compared with state-of-the-art models, our formulations significantly reduce computation times for most instances. In particular, for the Max-Total variants, 57 out of 60 instances can now be solved in less than a minute.

Several directions exist for future research. From a theoretical perspective, two interesting possibilities are deciding on the computational complexity of the Max-Total SDP and determining whether the AM model for PRDP is also a projection of GOM for that variant. Other ideas include the study of new facet-defining inequalities and the development of branching techniques based on geometric properties of the problem.

Finally, current methods still require large computation times to solve the Max–Min PRDP exactly, so there is still a lot of room for improvement.

References

[1] B. Dent, Cartography—Thematic Map Design, fifth ed., McGraw-Hill, 1999.
[2] T.A. Slocum, R.B. McMaster, F.C. Kessler, H.H. Howard, Thematic Cartography and Geographic Visualization, second ed., Prentice Hall, 2003.
[3] S. Cabello, H. Haverkort, M. van Kreveld, B. Speckmann, Algorithmic aspects of proportional symbol maps, Algorithmica 58 (3) (2010) 543–565.
[4] G. Kunigami, P.J. de Rezende, C.C. de Souza, T. Yunes, Generating optimal drawings of physically realizable symbol maps with integer programming, Vis. Comput. 28 (10) (2012) 1015–1026.
[5] G. Kunigami, P.J. de Rezende, C.C. de Souza, T. Yunes, Optimizing the layout of proportional symbol maps: Polyhedra and computation, INFORMS J. Comput. 26 (2) (2014) 199–207.
[6] G. Kunigami, Proportional symbol maps (Master's thesis) Institute of Computing, University of Campinas, Brazil, 2011.
[7] R.G. Cano, G. Kunigami, C.C. de Souza, P.J. de Rezende, A hybrid GRASP heuristic to construct effective drawings of proportional symbol maps, Comput. Oper. Res. 40 (5) (2012) 1435–1447.
[8] G. Nivasch, J. Pach, G. Tardos, The visible perimeter of an arrangement of disks, in: W. Didimo, M. Patrignani (Eds.), Proceedings of the 20th International Symposium on Graph Drawing, in: Lecture Notes in Computer Science, vol. 7704, Springer-Verlag, 2013, pp. 364–375.
[9] M. Grötschel, M. Jünger, G. Reinelt, On the acyclic subgraph polytope, Math. Program. 33 (1) (1985) 28–42.
[10] R. Müller, On the partial order polytope of a digraph, Math. Program. 73 (1996) 31–49.
[11] E. Balas, M. Oosten, On the dimension of projected polyhedra, Discrete Appl. Math. 87 (1–3) (1998) 1–9.
[12] G. Kunigami, R.G. Cano, P.J. de Rezende, C.C. de Souza, Proportional Symbol Maps—Benchmark Instances, 2011. www.ic.unicamp.br/~cid/Problem-instances/Symbol-Maps.