Central European Journal of Operations Research (2022) 30:1129–1149
https://doi.org/10.1007/s10100-021-00753-6

ORIGINAL PAPER

Optimized location of light sources to cover a rectangular region

Kristóf Kovács[1] · **Boglárka Tóth**[2]

Accepted: 15 May 2021 / Published online: 9 June 2021

Abstract
The problem we considered was proposed by an industrial partner. The aim is to locate light sources around a rectangular field such that the areas these illuminate cover the whole field. We assume these illuminated areas to be rectangular as well, parallel to the field. Covering an area with multiple lights is allowed. There are several types of light sources, priced differently with different sizes of their illuminated area. We aim to minimize the cost of the cover. We propose a constraint generation approach for solving this covering problem. We formulate a MIP model to locate the light sources such that a finite number of predetermined points have to be covered. The result does not necessarily solve the original problem, i.e. it does not cover the whole field. Therefore, a constraint generation model is built to calculate a non-covered point such that the first model has to improve its previous solution to cover this new point as well. If no uncovered point is found, the result is an optimal covering, thus we stop. We have also designed some set of additional constraints to exclude symmetrical solutions, to speed up the algorithm. We analyze the efficiency of the additional constraints and report some computational results in realistic settings.

Keywords Location · Covering problem · MIP · Constraint generation

1 Introduction

We aim to locate light sources around a rectangular field. The problem comes from an industrial application, where sports players are taught movement formations. Each

✉ Kristóf Kovács
kkovacs@math.bme.hu

Boglárka Tóth
boglarka@inf.szte.hu

[1] Dept. of Differential Equations, Budapest University of Technology and Economics, Budapest, Hungary

[2] Dept. of Computational Optimization, University of Szeged, Szeged, Hungary

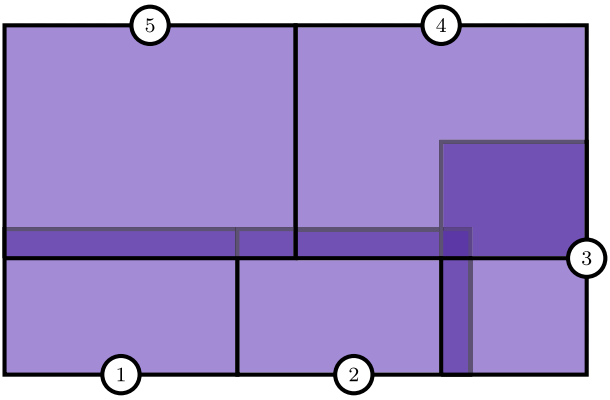

Fig. 1 An example of a feasible solution to our rectangle covering problem

player follows a path highlighted by a laser. The lasers are powerful enough to highlight multiple paths, so only their coverage of the field is important. We deal only with the location of the lights' sources knowing the area that they can illuminate. Of course, other applications are also possible, although the shape of the covering areas as rectangles has to be justified.

The light sources can only be placed outside the field. Different types of light sources are available, characterized by their lighted areas and their costs. These lighted areas are assumed to be rectangles to simplify the model and to ignore the light areas with a too oblique angle of incidence. The whole field has to be covered by the lighted rectangles.

Mathematically the problem can be formulated as a rectangle covering problem, where the covering items are rectangles themselves and they each have a specific side that has to be on, or outside the field (the one with its light source). An example of a feasible solution with two different types of lights can be seen in Fig. 1, where the light sources are represented by a numbered small circle. The blue rectangles represent the lighted areas with the darker shades being their overlapping areas.

Covering problems are very common in optimization, as they have many applications in computer graphics, artificial intelligence, and in the industry. In general, a covering problem is to find a minimal cover of a given set (e.g. a polygon) by some mold (e.g. rectangles). In Franzblau and Kleitman (1984) the authors introduce an algorithm to find the minimal cover of a rectilinear polygon by rectangles. The rectangles can be arbitrarily sized and overlapping is allowed, but all rectangles must be wholly contained in the polygon. The objective is to find a cover with the least amount of rectangles.

Iacob et al. (2003) solve a problem, where a polygon has to be covered with rectangles. They tried two objective functions, one where the number of rectangles used to cover is minimized, and another where the overlapping areas are minimized.

The Manhattan p-center problem can also be seen as a covering problem by squares, where a set of points has to be covered by p equal-sized squares, minimizing the size of the congruent covering squares. Drezner (1987) gives an $\mathcal{O}(n \log n)$ algorithm for

the rectangular (Manhattan) 3-center problem. A general algorithm for the p-center problem also exists, although far less efficient, see Ko et al. (1990).

A related problem of covering is the packing problem. Huang and Korf (2013) study a rectangle packing problem, where the objective is to find the smallest enclosing rectangle to contain a given set of rectangles without overlap. Similar problems appear frequently in computer graphics, as compressing and using a single larger graphics file is more efficient than using many smaller ones. Nöll and Strieker (2011) describe an efficient algorithm for packing charts into a texture atlas.

Dosa et al. (2020) define the board packing problem, where a rectangular board has to be covered partially by a set of rectangles. The board corresponds to some physical area where investments are to be made, and the rectangles correspond to potential investments which can give profit. Thus, the rectangles cannot overlap and the board does not have to be fully covered. They maximize the profit of the investment and solve the problem by a binary IP.

Surprisingly, although the above papers deal with problems similar to ours, the solution methods offered cannot be used here. We could not find any literature on more related problems. Thus, we have built new models to solve our specialized rectangle covering problem.

The contributions of the paper are the basic models for the constraint generation approach described in Sect. 2 and all the advanced constraints which exclude symmetrical solutions and force the rectangles to cover the border of the field, reported in Sect. 3. These modeling techniques can be useful in many problems, where symmetry gives the main difficulty for the models and methods. The efficiency of the models is shown in some industrial problems in Sect. 4. Finally, we draw our conclusions in Sect. 5.

This paper is based on the extended abstract published in Kovács and G.-Tóth (2019), which only contains an earlier version of the basic model described next.

2 Model description

Let us introduce our problem formally. The given data are the rectangular field with dimensions (a, b), $a \geq b$ and a finite set T of light source types. Set T contains triplets $((u_k, v_k), g_k)$, where (u_k, v_k) are the dimensions of the illuminated region of a light source k, and g_k is its price. Based on the application, we assume that $u_k, v_k \leq b$. The light source is always located at the longer side with dimension $u_k (\geq v_k)$. Each type of light source can be used as many times as needed, but the maximum number of light sources that can be used is n. This can be calculated by the given data, namely, the number of the smallest rectangles able to cover the field. With these parameters, the objective is to cover the whole field minimizing the overall cost of the light sources.

We made several attempts to describe the problem with a single Mixed-Integer Linear Programming (MILP) model with no success. Our technique, named the constraint generation approach, instead of covering the whole field, aims to cover an initially small set of points P spread across the field. This will be done by the main model called the Covering model. By covering a finite set of points we may not ensure that the whole field will be covered. Thus, an additional model generates a new point

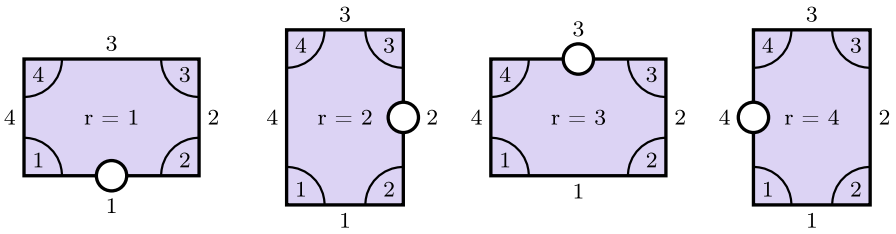

Fig. 2 Visual explanation of the rotation, corner and edge indexes

to be included in P, which was not covered by the solution of the main model. We call the latter one the Point Generation model. These two models are executed in a loop until a new non-covered point can not be found and thus the problem is solved.

An intermediate solution of the Covering model can have many symmetric solutions. Not dealing with these we might have to generate 3 or more points before a given arrangement of light sources is eliminated. To deal with these symmetries and other possible problems that may slow the solution process, in Sect. 3 we will introduce some set of constraints to improve the main Covering model.

To describe the main model, we start by introducing the notations (see the"Appendix" for a compiled list of notations), visualizing some of them in Figs. 2 and 3. The used indexes are fixed to the set they are iterating on. Namely, $r \in R = \{1, 2, 3, 4\}$ will always be used for the 4 rotations (it can also be interpreted as the 4 sides of the field), $k \in K = \{1, \ldots, |T|\}$ for the types of light sources and $c \in C = \{1, 2, 3, 4\}$ for the corners (or sides in some cases) of a rectangle from the bottom-left corner in counter-clockwise order. In Fig. 2 we show these notations, where the white circle represents the light source and the shaded rectangle shows its lighted area. For each rotation r the numbering of the corners and edges are written at the corners and on the sides of the rectangles. Additionally, we will use i, j as indices of possible light sources, $i, j \in L = \{1, \ldots, n\}$, and p as the index of points in the field to be covered, $p \in P = \{1, 2, \ldots\}$.

As shown in Fig. 3, for each rotation, type and corner of a light source it is easy to set a parameter dx_{rkc}, dy_{rkc} as the translations of corner c of light type k from the light source's position with rotation r. The coordinates of a point p to be covered are denoted by xp_p, yp_p, being parameters to the main Covering model.

The continuous variables x_i, y_i and binary variables τ_{irk} describe the light sources. Since we are locating the light sources on the field's borders, one position variable would be enough to describe a given position instead of the 2 dimensional coordinate (x_i, y_i). Using one position variable reduces the number of variables in the simplest model. However, in the more advanced model describing the constraints becomes much harder and so the computational effort does not decrease overall. The rotation and type of light source i is represented by τ_{irk}, whose value is 1 only if light source i has rotation r and type k.

In Fig. 3 the white circle containing 1 represents light source 1 with its coordinates (x_1, y_1). The dashed lines are the borders of the field. The dummy variables xc_{ic}, yc_{ic} serve as abbreviations for corner c of i's lighted rectangle.

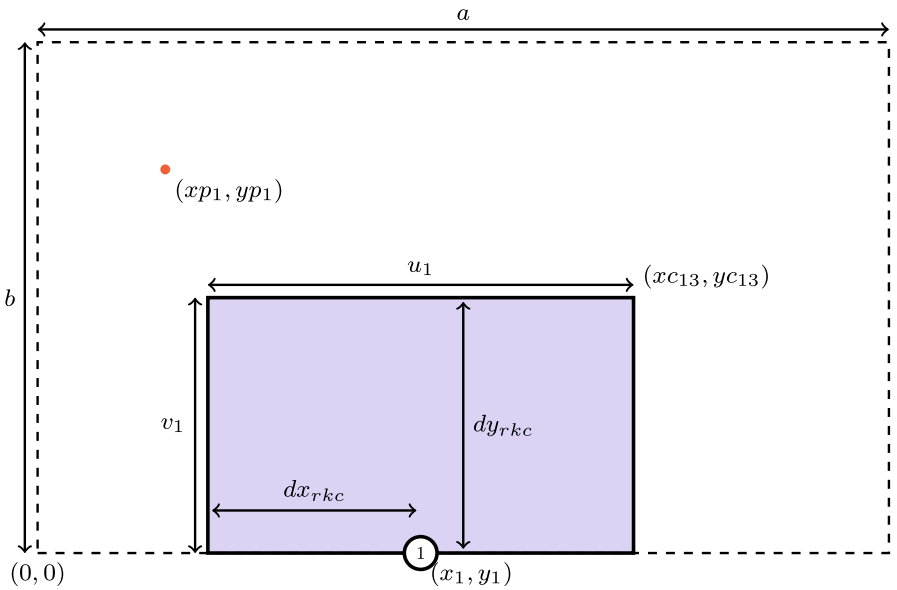

Fig. 3 Visual explanation of the model's parameters

Our objective function is to minimize the total price of the light sources,

$$\min \sum_{irk} g_k \tau_{irk}. \tag{1}$$

Let us now describe the constraints of the model. To ensure every light source has only one rotation and one type, or it is not used at all we have

$$\sum_{r,k} \tau_{irk} \leq 1 \quad \forall i \in L. \tag{2}$$

We set up the dummy variables xc_{ic}, yc_{ic} using the translation values dx_{rkc} and dy_{rkc} (see Fig. 3), which are predefined based on the parameters of the light source types (u_k, v_k),

$$xc_{ic} = x_i + \sum_{r,k} dx_{rkc} \tau_{irk} \quad \forall i \in L, c \in C \tag{3}$$

$$yc_{ic} = y_i + \sum_{r,k} dy_{rkc} \tau_{irk} \quad \forall i \in L, c \in C. \tag{4}$$

The following constraints ensure that each light source is located on the border of the field, where M is a sufficiently large number

$$0 \leq x_i \leq a, \quad 0 \leq y_i \leq b \qquad \forall i \in L \tag{5}$$

$$b - \left(1 - \sum_k \tau_{i3k}\right) M \leq y_i \leq 0 + \left(1 - \sum_k \tau_{i1k}\right) M \quad \forall i \in L \tag{6}$$

$$a - \left(1 - \sum_k \tau_{i2k}\right) M \leq x_i \leq 0 + \left(1 - \sum_k \tau_{i4k}\right) M \quad \forall i \in L \tag{7}$$

When implementing these constraints, choosing the tightest M can be beneficial to the optimization process. In general, the tightest choice of M is a or b depending on the coordinate appearing in the constraint. When this is not the case we will specify the used value when describing the constraints.

To further simplify the problem we constrain each rectangle to be wholly inside the field,

$$0 \leq xc_{i1}, yc_{i1} \quad \text{and} \quad xc_{i3} \leq a \quad \text{and} \quad yc_{i3} \leq b \qquad \forall i \in L. \tag{8}$$

To ensure that the points in P are covered, we introduce the binary variables ρ_{ip}, which is 1 if light source i is covering point p. The following constraints guarantee that at least one rectangle is covering every point in P.

$$\sum_i \rho_{ip} \geq 1 \qquad \forall p \in P \tag{9}$$

$$xc_{i1} - (1 - \rho_{ip})M \leq xp_p \leq xc_{i3} + (1 - \rho_{ip})M \qquad \forall i \in L, p \in P \tag{10}$$

$$yc_{i1} - (1 - \rho_{ip})M \leq yp_p \leq yc_{i3} + (1 - \rho_{ip})M \qquad \forall i \in L, p \in P \tag{11}$$

However, we have to ensure that only the light sources that are used can cover the points in P, so

$$\sum_p \rho_{ip} \leq |P| \sum_{r,k} \tau_{irk} \quad \forall i \in L. \tag{12}$$

The only initial element of P is the midpoint of the field.

To speed up the early iterations of the algorithm we implement a constraint to have the sum of the area of covering rectangles be at least as much as the area of the field

$$ab \leq \sum_{irk} \tau_{irk} \left(u_k v_k\right). \tag{13}$$

Lastly, we introduce the parameter LB which is updated after the first iteration to be the objective value of the last solved problem. Now we can formulate the following constraint on the objective function (1):

$$LB \leq \sum_{irk} \tau_{irk} g_k, \tag{14}$$

where we aim to help the optimizer forcing the solution to have at least as high objective value as the previous solution.

Let us move to the model in which either a new uncovered point is located, or we prove that the whole field is covered.

2.1 Point generation model

We aim to find a new point that is not covered if one exists. We have found that the best way is to seek a point maximizing the minimum ℓ_∞ distance from all the rectangles. In this way, the new point with coordinates xn, yn is more or less centralized in the uncovered area. However, in many cases, the optimal region of this objective function is a segment instead of a point. In practice, it is desirable to choose the new point to be close to the midpoint of the field. Thus, we add that to the objective function with a very small weight. The objective is then

$$\max d - 0.001(dxm + dym) \tag{15}$$

where d is the minimal ℓ_∞ distance of the new point from all the rectangles and $dxm + dym$ is the new point's ℓ_1 distance from the center of the field.

We have just a few constraints in this model to set the distance variables. The binary variables θ_{ij} indicate if the new point is inside of light source i's half-plane corresponding to its side j. The following constraints define θ_{ij} and also the minimal distance d. If the new point is outside j's half-plane, then $\theta_{ij} = 0$ and d is less than the distance from that side, otherwise, the constraint is not in effect. For instance, let us examine (16). The second part ensures that $\theta_{i2} = 0$ if the distance $xc_{i1} - xn$ is positive, and so the first part guarantees $d \leq xc_{i1} - xn$. If $xc_{i1} - xn$ is negative, only the optimization (maximizing d) will cause $\theta_{i2} = 1$, otherwise the distance could get a negative value. The best choice for M is a or b, as usual. The last constraint ensures that the new point cannot be inside any rectangle, as at least one θ_{ij} must be 0 for all the rectangles.

$$d \leq xc_{i1} - xn + \theta_{i2}M \leq M \qquad \forall i \in L \tag{16}$$
$$d \leq xn - xc_{i3} + \theta_{i4}M \leq M \qquad \forall i \in L \tag{17}$$
$$d \leq yc_{i1} - yn + \theta_{i3}M \leq M \qquad \forall i \in L \tag{18}$$
$$d \leq yn - yc_{i3} + \theta_{i1}M \leq M \qquad \forall i \in L \tag{19}$$
$$\sum_j \theta_{ij} \leq 3 \qquad \forall i \in L \tag{20}$$

Lower bounds on the coordinate distances from the midpoint are set by the constraints

$$a/2 - xn \leq dxm \geq xn - a/2, \tag{21}$$
$$b/2 - yn \leq dym \geq yn - b/2. \tag{22}$$

The objective (15) ensures that dxm, dym will not be higher than the real distance.

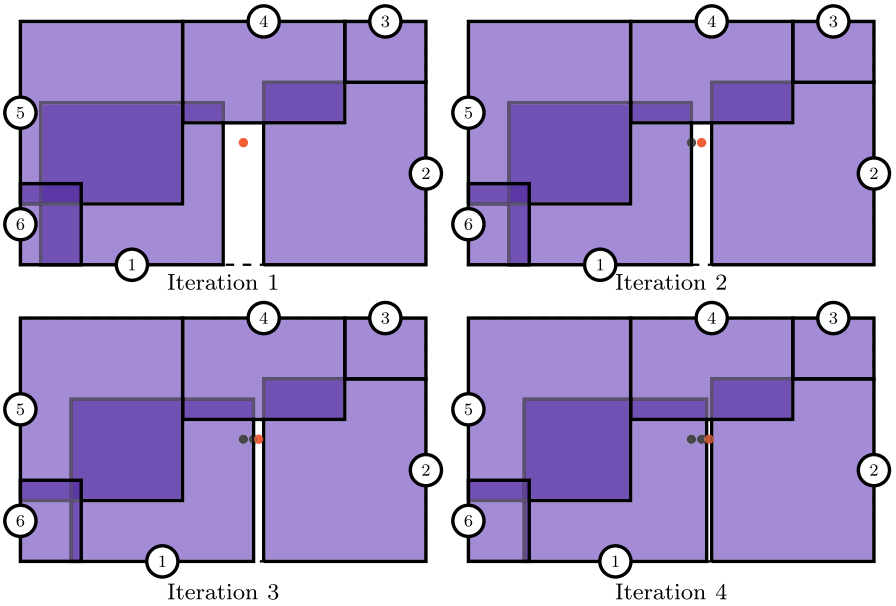

Fig. 4 Four iterations to close the gap between two rectangles. The red dots are generated by the point generation model and depicted by black when covered

If the new point is found with $d > 0$, this point is added to the set P, and the main Covering model is solved again until no such point is found. In the latter case, $d \leq 0$ ensures that the last solution covers the whole field, so we can stop the algorithm.

It is worth mentioning that the point generation model takes up an insignificant percentage of the total execution time. Thus, in Sect. 3 we focus on improving the main Covering model only. Before that we describe an alternative formulation where we combine the coordinate variables x_i, y_i into one variable, hoping to achieve improvements in the solution time.

3 Improving the main covering model

The model described in the previous section can solve our problem to optimality. However, this can be inefficient due to the possible appearance of partial solutions that are just symmetric versions of a previous solution. Also, neighboring rectangles are not ensured to touch or overlap. These make it hard for the algorithm to plug a gap between two rectangles as only half of the gap might be eliminated in each iteration. An example can be seen in Fig. 4. In this section, we try to alleviate these problems by introducing new constraints to the main Covering model.

3.1 Order on the rectangles

The following constraints ensure that the rectangles are ordered, so we can construct constraints that refer to the neighbors of a given rectangle. It also allows us to eliminate

symmetric solutions, as well as ensure that the outline of the field is covered in every partial solution. We introduce the notation $n(o)$ ($p(o)$) for the next (previous) index after (before) o in its given set O in a circular manner. In other words $n(o) = o + 1$ if $o < |O|$, and $n(|O|) = 1$.

We create the order of the rectangles in four steps, see Fig. 5 for the result including these steps one by one. First, we ensure that no unused light source(s) can appear between any two used light sources. In other words, if there are z used light sources, the light sources $i = 1, \ldots, z$ are the used ones, while $(z + 1), \ldots, n$ are the unused ones. The following constraint ensures all $\tau_{irk} = 0$ for $i > z$, as the right-hand-side (RHS) is always z:

$$i \sum_{r,k} \tau_{irk} \leq \sum_{j,r,k} \tau_{jrk} \quad \forall i \in L. \tag{23}$$

Next, an order on the edges of the field is created. The constraint

$$\sum_{k} \tau_{11k} \geq \sum_{i,k} \frac{\tau_{i1k} + \tau_{i3k}}{n+1} \tag{24}$$

ensures that if there is a light source either on the top or on the bottom edge of the field (edge 1 and 3, see Fig. 2), then the rectangle with the first index, $i = 1$, will be on the bottom edge (see Fig. 5b). The next constraint ensures that rectangles with the same rotation are grouped. In other words, it makes sure that the next rectangle has a rotation equal or higher (see Fig. 5c). The constraint ignores unused rectangles, as the RHS is 4 for an unused $n(i)$, because its sum for $\tau_{n(i),rk}$ gives 0 in that case, thus making the constraint ineffective. Leaving out the $i = n$ case, we do not force the first rectangle to have a higher rotation than the last.

$$\sum_{r,k} r\tau_{irk} \leq \sum_{r,k} r\tau_{n(i),r,k} + 4\left(1 - \sum_{r,k} \tau_{n(i),rk}\right) \quad \forall i \in L, i < n \tag{25}$$

Now all that is left is to fix the order on each edge (see Fig. 5d). We ordered those with one set of constraints for each side, namely

$$x_i - M\left(1 - \sum_{k} \tau_{i1k}\right) \leq x_{n(i)} + M\left(1 - \sum_{k} \tau_{n(i),1,k}\right) \quad \forall i \in L, i < n \tag{26}$$

$$y_i - M\left(1 - \sum_{k} \tau_{i2k}\right) \leq y_{n(i)} + M\left(1 - \sum_{k} \tau_{n(i),2,k}\right) \quad \forall i \in L, i < n \tag{27}$$

$$x_i + M\left(1 - \sum_{k} \tau_{i3k}\right) \geq x_{n(i)} - M\left(1 - \sum_{k} \tau_{n(i),3,k}\right) \quad \forall i \in L, i < n \tag{28}$$

$$y_i + M\left(1 - \sum_{k} \tau_{i4k}\right) \geq y_{n(i)} - M\left(1 - \sum_{k} \tau_{n(i),4,k}\right) \quad \forall i \in L, i < n. \tag{29}$$

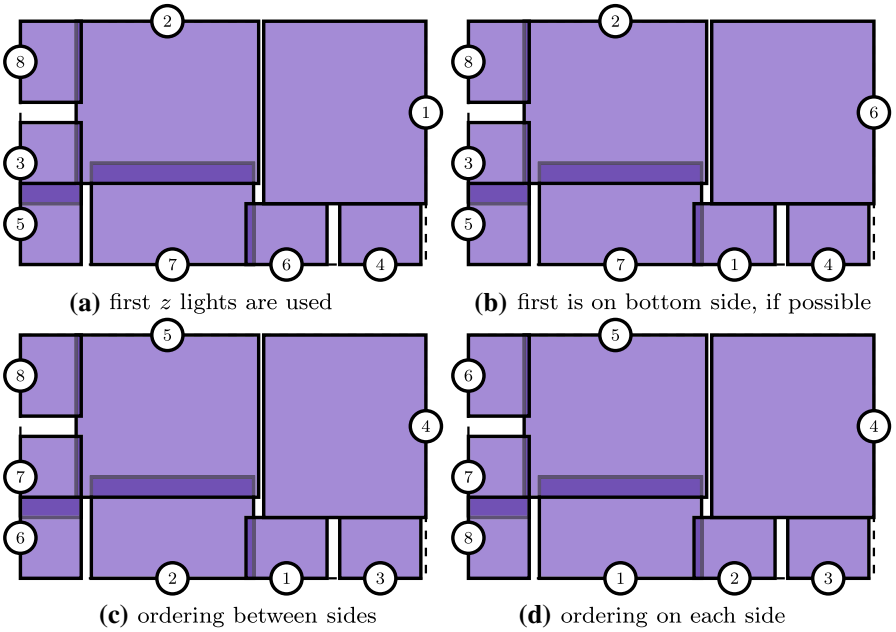

Fig. 5 The evolution of the order with each step of constraints

Here the big M parts disappear only if both i and $n(i)$ are on the same edge, so the constraint defines the relation on rectangles on the same edge. When either of the big Ms remain in the constraint (so at least one of the light sources is not on the given edge), the constraint becomes nonbinding for all coordinate values. The usual choice of a or b for the big M is still valid here.

3.2 Eliminating symmetries

Now that we have an order on the rectangles we can define symmetric placements for a given order of rectangle types and eliminate them from all future solutions of the main model. An example of a solution and its vertical mirror can be seen in Fig. 6. We introduce a new index and set $f \in E$, as the index and set of eliminations. Set E will be growing after each execution of the main Covering model. The parameter e_{firk} holds the information for the order of rectangle types that should be eliminated for each $f \in E$.

For each new elimination $\hat{f}$ describing a symmetric solution, let $E = E \cup \{\hat{f}\}$. A new set of parameters is created for this $\hat{f}$, where we set each $e_{\hat{f}irk} = \frac{1}{\hat{z}-0.5}$ if $\tau_{irk} = 1$ in the arrangement, $e_{\hat{f}irk} = 0$ otherwise. Here $\hat{z}$ is the number of active light sources for this particular solution. The parameters $e_{\hat{f}irk}$ will have the property

$$\sum_{i,r,k} e_{\hat{f}irk} = \frac{\hat{z}}{\hat{z}-0.5} > 1 \qquad \sum_{i,r,k} e_{\hat{f}irk} - \frac{1}{\hat{z}-0.5} = \frac{\hat{z}-1}{\hat{z}-0.5} < 1.$$

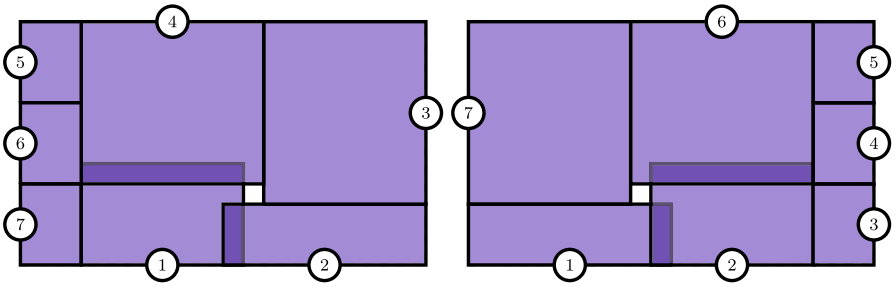

Fig. 6 A partial solution and its vertical mirror

This way the following constraint prevents all arrangements contained in E

$$\sum_{k} \tau_{11k} + \sum_{i,r,k} \tau_{irk} e_{firk} \leq 2 \quad \forall f \in E, \tag{30}$$

unless the first rectangle is not on the first side. In general, as the first sum is 1 (having the first light source on the first side), the second sum is greater than 1 only if it has the structure of a symmetric solution. In this case, it does not fulfill the constraint and will not be considered for the solution. If at least one rectangle differs in the structure, the second sum becomes less than 1, thus fulfilling the constraint. When the first light source is not on the first side, we have a special case, as constraint (24) ensures that this only happens when there are no rectangles on sides 1 and 3. A solution like that has to use light sources that extend their lighted area to the half of the field's width a. These solutions usually consist of only two rectangles covering the whole field and thus can safely be excluded from the symmetry calculations.

After each execution of the main model, we extend set E by all the symmetric or rotated versions of the current solution (at most 3). We leave the current solution as the only feasible solution with its arrangement of rectangles.

3.3 Forcing neighbors

The last method we use is the most complicated but also the most rewarding in terms of saving computational time as will be evident from Sect. 4. The idea is that every proper cover of the field has to cover each edge of the field as well. Thus by ensuring that each edge is individually covered we can speed up the convergence to a whole field cover. Since we have an order on the rectangles we can use that to ensure that neighboring rectangles are tight and thus cover the edges of the field.

The complicated part of these constraints is that a given rectangle can cover not just the side on which its light source is, but neighboring sides as well if it is in a corner. Thus the first few constraints introduce variables that indicate if a given rectangle touches a given side of the field and if it's in one of the field's corners.

The following constraints give an upper bound to the binary variable ψ_{io} which is an extension of τ_{irk} that shows if light source i touches side o of the field or not:

$$M\,(1-\psi_{i1}) \geq yc_{i1} \quad \forall i \in L \tag{31}$$
$$a\psi_{i2} \leq xc_{i2} \quad \forall i \in L \tag{32}$$
$$b\psi_{i3} \leq yc_{i3} \quad \forall i \in L \tag{33}$$
$$M\,(1-\psi_{i4}) \geq xc_{i4} \quad \forall i \in L. \tag{34}$$

The following constraint ensures that every active rectangle is on at least 1 side.

$$\sum_{o} \psi_{io} \geq \sum_{rk} \tau_{irk} \quad \forall i \in L. \tag{35}$$

Next, we set up the binary variable σ_{io} such that it shows if light source i is in corner o of the field. The order of the sides and corners are set up so that the edges of corner o are $p(o)$ and o, as can be seen in Fig. 2.

$$\psi_{i,p(o)} + \psi_{i,o} - 1 \leq \sigma_{i,o} \quad \forall i \in L, o \in R \tag{36}$$
$$\psi_{i,p(o)} \geq \sigma_{i,o} \quad \forall i \in L, o \in R \tag{37}$$
$$\psi_{i,o} \geq \sigma_{i,o} \quad \forall i \in L, o \in R \tag{38}$$

(36)–(38) are McCormick linearizations of the relations $\psi_{i,p(o)}\psi_{i,o} = \sigma_{i,o} \quad \forall i \in L, o \in R$, i.e. a rectangle corner is on the field's corner if and only if the corresponding two sides are on the field's sides.

Using these binary variables we can force proper order on the rectangles near corners. We still need to make sure that only active rectangles are considered, so

$$\sum_{r,k} \tau_{irk} \geq \sigma_{ic} \quad \forall i \in L, c \in C, \tag{39}$$

and that only one rectangle is in each corner,

$$\sum_{i} \sigma_{ic} = 1 \quad \forall c \in C. \tag{40}$$

Note that a rectangle at a corner is not constrained to have two $\psi_{io} = 1, o \in R$ variables (only upper bounds are given for them in (31)–(34)). Thus, constraint (40) is not effective until we make sure that a rectangle is in a corner iff the corresponding $\sigma_{ic} = 1$. Otherwise, there could be more than one rectangle in a corner for which the conditions (45)–(52) would not be effective. Here we use a small ε instead of a big M to make sure there is a minimum distance from the corner for all but one rectangle.

$$\epsilon\,(1-\sigma_{i1}) \leq xc_{i1} + yc_{i1} \qquad \forall i \in L \tag{41}$$
$$\epsilon\,(1-\sigma_{i2}) \leq a - xc_{i2} + yc_{i2} \qquad \forall i \in L \tag{42}$$

$$\epsilon\,(1-\sigma_{i3}) \le a - xc_{i3} + b - yc_{i3} \qquad \forall i \in L \tag{43}$$
$$\epsilon\,(1-\sigma_{i4}) \le xc_{i4} + b - yc_{i4} \qquad \forall i \in L \tag{44}$$

Now that we have set up the variables, we can force neighboring rectangles to be next to each other (either touching or overlapping but within the order). This way, the edges of the field will be covered automatically. We also bring more structure to a feasible solution, reducing the average number of iterations required to rule out a given arrangement of rectangles.

We define two constraints for each side, namely

$$xc_{i3} \ge xc_{n(i),1} - \left(\sum_{r,k}\tau_{irk} + \sum_{r,k}\tau_{n(i),r,k} - \psi_{i1} - \psi_{n(i),1}\right) M \qquad \forall i \in L \tag{45}$$
$$yc_{i3} \ge yc_{n(i),1} - (2 - \psi_{i2} - \psi_{n(i),2})M \qquad \forall i \in L \tag{46}$$
$$xc_{i1} \le xc_{n(i),3} - (2 - \psi_{i3} - \psi_{n(i),3})M \qquad \forall i \in L \tag{47}$$
$$yc_{i1} \le yc_{n(i),3} - \left(\sum_{r,k}\tau_{irk} + \sum_{r,k}\tau_{n(i),r,k} - \psi_{i4} - \psi_{n(i),4}\right) M \qquad \forall i \in L \tag{48}$$
$$xc_{i3} \le xc_{n(i),3} - 0.01 + (2 - \psi_{i1} - \psi_{n(i),1})M \qquad \forall i \in L \tag{49}$$
$$yc_{i3} \le yc_{n(i),3} - 0.01 + (2 - \psi_{i2} - \psi_{n(i),2})M \qquad \forall i \in L \tag{50}$$
$$xc_{i3} \ge xc_{n(i),3} + 0.01 - (2 - \psi_{i3} - \psi_{n(i),3})M \qquad \forall i \in L \tag{51}$$
$$yc_{i3} \ge yc_{n(i),3} + 0.01 - (2 - \psi_{i4} - \psi_{n(i),4})M \qquad \forall i \in L. \tag{52}$$

First, let us examine constraints (46) and (50), with two consecutive rectangles on side 2 (so the big M parts vanish). These constraints force the 3rd corner of rectangle i to be between the 1st and 3rd corner of rectangle $n(i)$. That is, no hole between them, only intersection or touch is allowed. When two consecutive rectangles are not on side 2, the big M part makes these constraints ineffective. The other pair of constraints can be explained similarly, although for sides 1 and 4 we need to take care of the inactive rectangles. The next constraints put the corners of the inactive light sources on top of the first light source's 4th corner. This way, the inactive rectangles function like a reference to the first rectangle.

$$xc_{14} - M\sum_{rk}\tau_{irk} \le xc_{ic} \le xc_{i4} + M\sum_{rk}\tau_{irk} \quad \forall i \in L, c \in C \tag{53}$$
$$yc_{14} - M\sum_{rk}\tau_{irk} \le yc_{ic} \le yc_{i4} + M\sum_{rk}\tau_{irk} \quad \forall i \in L, c \in C. \tag{54}$$

Now, if we look back at the multiplier of the big M in either (45) or (48), we can see that it permits one of the rectangles to not be on the given side if one of the rectangles is inactive. This way, we can connect the last rectangle using the inactive rectangles as proxies to the first rectangle. Constraint (45) corresponds to the case when the last

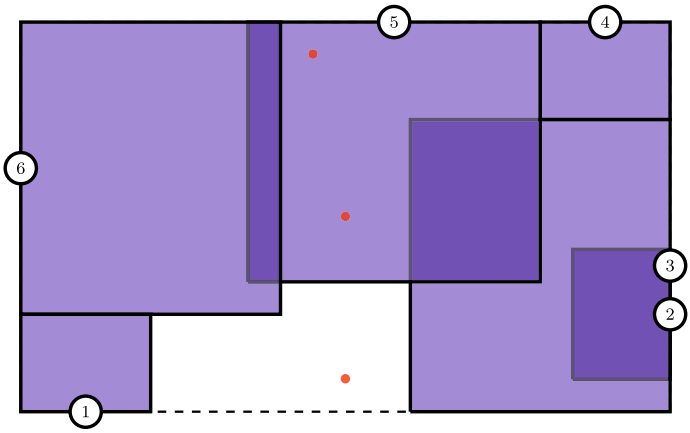

Fig. 7 A feasible partial solution without constraints (55) and (56)

rectangle is in corner 1, while constraint (48) fits the case when the first rectangle is in corner 1.

Lastly, we still have a special case to exclude, which can be seen in Fig. 7. What happens here is that the order on each edge is appropriate, the neighboring in each edge is proper as well, except that light source 2 should be in corner 2 since it comes before light source 3. Thus, the constraints forcing the neighbors are invalidated since light source 1 and light source 2 do not share the same edge.

To fix this, suppose there is a rectangle i in corner o which touches side $p(o)$ and o (in this order). The preceding rectangle $p(i)$ cannot be on side o. The next rectangle $n(i)$ cannot be on side $p(o)$. For example, in Fig. 7 rectangle 2 should not be on side 3, since rectangle 3 is in corner 2, which should prohibit the previous rectangle to be on side 3. The next constraints describe this rule.

$$1 - \psi_{p(i),o} \geq \sigma_{i,o} \quad \forall i \in L, o \in R \tag{55}$$

$$1 - \psi_{n(i),p(o)} \geq \sigma_{i,o} \quad \forall i \in L, o \in R. \tag{56}$$

The effects of these additional constraints compared to the basic model can be seen in the next section, where we discuss the computational results obtained with this constraint generation approach.

4 Computational results

We used AMPL (Fourer et al. 2002) to implement the models and solved them with CPLEX 12.8.0 (cpl 2020). The tests were run on a machine with 8 GB of memory and an Intel Core i7-4710HQ CPU. The execution times (in seconds) can be seen in Table 1. The tests were run for different sets of light sources with $|K| = 2, 3, 4, 5, 7$, and with $n = 10$, the maximum number of available light sources.

Table 1 Execution times (in seconds) of different sets of constraints for maximum 10 light sources

Problem	#light types	Basic	OE	OF	OEF	Optimum
1	2	22	209	24	38	16
2	3	61	346	15	20	14
3	5	>1800	318	19	13	13
4	7	>1800	738	29	27	13
5	2	>1800	>1800	278	187	6
6	4	1769	>1800	15	10	58
7	2	64	440	2	2	16
8	3	3	5	1	1	13
Average		>915	>708	48	38	

Table 2 Model iterations and MIP iterations of different sets of constraints for the constraint generation method

Problem	Basic		OE		OT		OET	
	Iter.	MIP iter	Iter.	MIP iter	Iter.	MIP iter	Iter.	MIP iter
1	58	261,510	61	6,745,648	9	1,450,598	7	1,403,607
2	83	724,743	93	9,803,603	7	551,204	6	832,790
3	41	28,210,833	69	8,952,183	4	649,549	2	489,818
4	39	31,558,783	80	16,705,287	6	782,319	5	589,346
5	95	7,153,738	103	34,537,365	29	7,142,275	26	4,742,987
6	204	12,765,301	105	19,084,889	3	804,672	2	526,814
7	64	302,940	100	4,754,435	1	100,613	1	100,613
8	20	5741	23	119,770	1	13,729	1	13,729
Average	76	10,122,949	79	12,587,898	8	1,436,870	6	1,087,463

Problems 1–4 were generated by incrementally adding new types of light sources to the previous problem, with different attributes. It is worth mentioning that adding new light types can make a problem easier or harder depending on their attributes. Problems 5–8 were designed to test the limits of the approach. In all problems the dimensions of the field were $a = 20$, $b = 12$.

Four configurations of the constraints were tested. The one named basic contains the model from Sect. 2, OE has additionally the Order constraints (23–reforderspsend) as well as the Elimination of symmetric solutions (30). OF includes constraints on the order (23–44) and on the field edge covering (45–52) apart from the basic model. Lastly, OEF has every constraint we listed for the constraint generation approach. These computational times can be seen in Table 1, while the number of model iterations and the sum of MIP iterations for the whole algorithm are shown in Table 2.

The results make it clear that the constraint configuration using all the improvements is superior in almost all cases to the others. It can also be seen that the most important constraints apart from the order (which in itself does not lead to any improvement) are the field edge covering constraints. The elimination method improves the consis-

Table 3 Execution time, model and MIP iterations of the OEF version for problem 4 as a function of n, the maximum number of light sources

	$n = 10$	$n = 11$	$n = 12$	$n = 13$	$n = 14$	$n = 15$
Time (s)	27	131	32	65	67	54
Iter.	5	13	1	7	6	3
MIP iter.	589,346	2,865,134	725,321	1,138,136	1,386,597	1,020,826

tency of the algorithm, but highly increases the execution times without the field edge covering constraints.

Looking at the number of model iterations the average number of iterations increases if the elimination is used without the field edge covering constraints. This can be attributed to the problem seen in Fig. 4. It is also worth mentioning that the method with all constraints (OEF) scales well with the maximum number of light sources as can be seen in Table 3. Thus, it is possible to verify if a higher number of light sources would lead to a better solution.

It is important to mention that comparing the models is difficult because there are many feasible placements of the rectangles for a given objective value. Depending on the starting solution the execution time and the number of iterations can highly vary, see Table 3.

5 Conclusions

In this paper, a new covering problem is formulated and solved, where a rectangle has to be covered by different types of rectangles that describe the illuminated area of light sources. As the problem comes from a new innovative industrial application, we did not find existing models or algorithms to solve this specific problem.

To build a useful model, a constraint generation approach is designed. The concept is to cover a set of points and check if the whole area is covered. If there is an uncovered area, we locate a new point in it and rerun the covering model requiring this new point to be covered as well. This approach is efficient for all the cases appearing in the industrial application. Still, we could improve it by introducing ordering on the rectangles, ensuring no gap between consecutive rectangles, and finally, eliminating symmetric solutions.

Apart from setting up these models to solve the given covering problem, we analyzed the model components. We checked the efficiency of the additional constraints. We have found that forcing the most structure on the rectangles, by having neighboring rectangles touch increases the model's consistency by a large margin. We also learned that eliminating symmetric solutions helps more if the model already forces a stricter structure on the rectangles.

As for future research, we plan to extend the model by allowing arbitrary rotations of the light sources, and generalizing the shape of the illuminated area to convex polygons. In the industrial application, certain areas should be covered by multiple light sources, leading to a more complex problem to be solved as a long-term objective.

Acknowledgements This research was supported by the Project "Deepening the activities of the Hungarian Industrial Innovation Mathematical Service Network HU-MATHS-IN", No. EFOP-3.6.2-16-2017-00015, and by the European Union. It was also co-funded by the European Social Fund. We would like to thank Tamás Vinkó and Benjamin Balogh for their useful comments on the paper.

Funding Open access funding provided by Budapest University of Technology and Economics.

6 Appendix

6.1 Notation for the base model

Indices

i, j	Index of the possible light sources, $i, j \in L = \{1, \ldots, n\}$
r	Type of rotation, $r \in R = \{1, 2, 3, 4\}$ (see Fig. 2)
k	Light source types $k \in K = \{1, \ldots, \lvert T \rvert\}$
c	Index of corners $c \in C = \{1, 2, 3, 4\}$ from bottom-left corner in counter clockwise order (see Fig. 2)
p	Index of points in the field to be covered $p \in P = \{1, \ldots, \}$

Parameters

a, b	Dimensions of the field
n	Maximum number of light sources, $n = \lvert L \rvert$
u_k, v_k	Dimensions of the lighted rectangle for light type k
g_k	price of light type k
dx_{rkc}, dy_{rkc}	Translations of corner c of light type k from the light source's position with rotation r
xp_p, yp_p	Coordinates of a predefined point p

Variables

x_i, y_i	Position of light source i
τ_{irk}	Binary variable to indicate the type and rotation of a given light source: $\tau_{irk} = 1$ if light source i has rotation r and type k
ρ_{ip}	Binary variables to indicate if point p is covered by light source i

Dummy variables

xc_{ic}, yc_{ic}	Position of corner c of light source i

6.2 Notation for the point generation model

Variables

xn, yn	Position of the new point to be covered, $xn \in [0, a]$, $yn \in [0, b]$
θ_{ij}	Binary variables to indicate if the new point is inside of light source i corresponding to its side j
d	Minimum ℓ_∞ distance from the new point to the closest rectangle
dxm, dym	Coordinate distances from the new point to the midpoint of the field

6.3 Notation for the improved model

Indices

$n(o)$	Next index after o in a circular manner for any set $O = \{1, \ldots, \lvert O\rvert\}$: $n(o) = o + 1$ if $o < \lvert O\rvert$, $n(o) = 1$ otherwise
$p(o)$	Previous index before o in a circular manner for any set $O = \{1, \ldots, \lvert O\rvert\}$: $p(o) = o - 1$ if $o > 1$, $p(o) = \lvert O\rvert$ otherwise
f	Index for the eliminated placements, $f \in E$

Parameters

ϵ	A small distance; a rectangle's corner is either at a corner of the field or cannot be ϵ-distance from said corner
e_{first}	This is used to eliminate the solutions that are symmetric appearances of earlier solutions

Variables

ψ_{io}	Binary variable to indicate if light source i is touching side o
σ_{ic}	Binary variable to indicate if light source i is in corner c of the field

6.4 Parameters for the test problems

All parameters of the test cases are given in Table 4. The optimal coverings are shown in Fig. 8.

Table 4 Parameters of each problem

Problem	u_1	v_1	g_1	u_2	v_2	g_2	u_3	v_3	g_3	u_4	v_4	g_4	u_5	v_5	g_5	u_6	v_6	g_6	u_7	v_7	g_7
1	4	3	1	10	8	4															
2	4	3	1	10	8	4	8	5	2												
3	4	3	1	10	8	4	8	5	2	9	8	3	10	11	6						
4	4	3	1	10	8	4	8	5	2	9	8	3	10	11	6	3	8	2	10	3	2
5	9	5	1	11	7	2															
6	4	4	6	6	6	10	4	6	7	6	2	1									
7	3	4	1	7	8	3															
8	4	4	1	8	8	3	4	6	2												

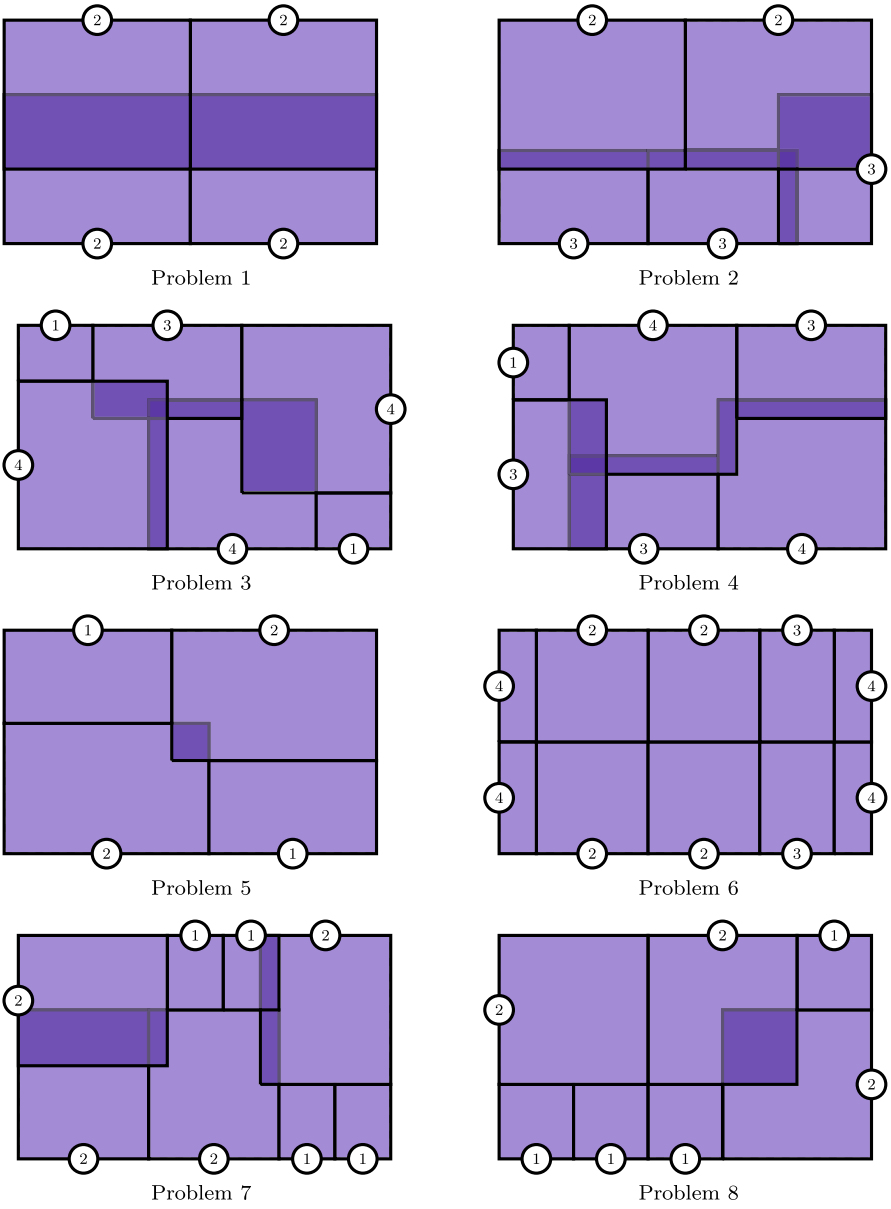

Fig. 8 Optimal solutions to each problem. The rectangles are labeled by their type (k) according to Table 4

References

Cplex optimizer (2020). https://www.ibm.com/analytics/cplexoptimizer

Dosa G, Hvattum LM, Olaj T, Tuza Z (2020) The board packing problem: packing rectangles into a board to maximize profit. In Pannonian conference on advances in information technology (PCIT 2020), pp 10–16

Drezner Z (1987) On the rectangular p-center problem. Naval Res Logist (NRL) 34(229–234):04
Fourer R, Gay DM, Kernighan BW (2002) AMPL: a modeling language for mathematical programming. Cengage Learning, ISBN 0534388094. https://ampl.com/resources/the-ampl-book/
Franzblau D, Kleitman D (1984) An algorithm for covering polygons with rectangles. Inf Control 63(3):164–189. ISSN 0019-9958. https://doi.org/10.1016/S0019-9958(84)80012-1
Huang E, Korf RE (2013) Optimal rectangle packing: an absolute placement approach. J Artif Int Res 46 (1):47–87. ISSN 1076-9757. https://doi.org/10.1613/jair.3735
Iacob P, Marinescu D, Luca C (2003) Covering with rectangular pieces. Analele Universitatii "Ovidius" Constanta - Seria Matematica, 11(2):75–86. https://www.anstuocmath.ro/mathematics/pdf6/75_86_PIacob_DMarinescu_CLuca.pdf
Ko MT, Lee RCT, Chang JS (1990) Rectilinear m-center problem. Naval Res Logist (NRL) 37(3):419–427. https://doi.org/10.1002/nav.3800370306
Kovács K, G.-Tóth B (2019) Rectangle covering. In: AIP conference proceedings, vol 2070, p 020036. https://doi.org/10.1063/1.5090003, https://aip.scitation.org/doi/abs/10.1063/1.5090003
Nöll T, Strieker D, (2011) Efficient packing of arbitrary shaped charts for automatic texture atlas generation. Computer Graphics Forum 30(4):1309–1317. https://doi.org/10.1111/j.1467-8659.2011.01990.x.https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2011.01990.x