Valid constraints for the Point Packing in a Square problem

Alberto Costa[a,*]

[a] *Singapore University of Technology and Design, 138682 Singapore*

Abstract

We consider the problem of placing n points in the unit square in such a way as to maximize their minimum pairwise distance m. Starting from two properties of the optimal solution presented by Locatelli and Raber in [Discrete Applied Mathematics **122**(1-3):139-166, 2002], and using the known theoretical lower and upper bounds, we derive some constraints for tightening the original formulation of the problem.

Keywords: symmetry, nonconvex NLP, point packing in a square

1. Introduction

Circle packing in a square is a classical problem in mathematics, with several applications, for example cutting problems [1, 2, 3] and container loading [4, 5]. Other examples can be found in [6].

This problem can be stated in different but equivalent ways, e.g., place non-overlapping circles in a square maximizing their common radius, or place points in a square maximizing their minimum pairwise distance. We consider the latter, which can be defined more formally as follows:

> Point Packing in a Square (PPS). Given an integer $n > 0$, place n points in the unit square $U = [0,1]^2$ such that their minimum pairwise distance m is maximal.

Let $N = \{i \in \mathbb{N} : 1 \leq i \leq n\}$. A mathematical programming formulation for the PPS problem is the following:

$$\max \quad \alpha \tag{1}$$

$$\text{s.t.} \quad \forall i \in N, \forall j \in N : i < j \quad x_i^2 + x_j^2 - 2x_i x_j + y_i^2 + y_j^2 - 2y_i y_j \geq \alpha \tag{2}$$

$$\forall i \in N \quad x_i \in [0,1] \tag{3}$$

$$\forall i \in N \quad y_i \in [0,1] \tag{4}$$

$$\alpha \in \mathbb{R}, \tag{5}$$

*Corresponding author
Email address: costa@sutd.edu.sg (Alberto Costa)

where $\alpha = m^2$; this variable should be defined as nonnegative, but we are maximizing it so we can let it be free. Note that this problem is difficult to solve because it is nonconvex and nonlinear due to the distance constraints (2). Actually, these constraints could alternatively be written in this way:

$$\forall i \in N, \forall j \in N : i < j \quad (x_j - x_i)^2 + (y_j - y_i)^2 \geq \alpha. \tag{6}$$

However, some experiments performed using the solver COUENNE [7] showed that for large instances it is better to employ (2).

In [8] some bounds for the optimal distance m^* are introduced (in order to have a uniform notation, we employ m^* to refer to the optimal distance for the PPS problem with n points, without reporting the index n):

$$L_n \leq m^* \leq U_n, \tag{7}$$

where

$$L_n = \sqrt{\frac{2}{\sqrt{3}n}}. \tag{8}$$

$$U_n = \frac{1}{n-1} + \sqrt{\frac{1}{(n-1)^2} + \frac{2}{\sqrt{3}(n-1)}}. \tag{9}$$

Using these bounds and some properties of the optimal solution presented in the next section, we introduce further constraints to tighten the formulation (1)-(5).

2. Improved formulation for the PPS problem

In this section we first report some properties of the optimal solution of PPS, whose proofs are provided in [9]. Starting from these properties, we prove a theorem which allows us to fix some points of the optimal solution along the left and right sides of the square. Then three corollaries of this theorem are presented, as well as additional constraints which are helpful to tighten the formulation. The improved PPS formulation is reported at the end of the section.

2.1. Fixing points along sides of the square

In [10] the authors present two properties for the PPS problem (in their paper the name of the problem is PSP), which are proved in [9]:

Property 1 (Locatelli and Raber [10])**.** *There always exists an optimal solution of the PPS problem such that at each vertex v of the unit square U, incident to the sides e_1 and e_2, one and only one of the following statements holds:*

(1a) a point of the optimal solution is located at the vertex v;

(1b) two points of the optimal solution belong to the sides e_1 and e_2 and have distance equal to the optimal one.

Property 2 (Locatelli and Raber [10])**.** *There always exists an optimal solution of the PPS problem such that along each side of U there is no portion of the side of width greater than or equal to twice the optimal distance m^*, which does not contain any point of the optimal solution.*

Using the Property 1 we can prove the following lemma (the proof can also be found in [11], and a short version is presented in [12]):

Lemma 1. *Consider the PPS problem with $n \geq 4$. There is always an optimal solution where at least two points are on the left side of the square, and at least two points are on the right side of the square.*

Proof. Consider the left side of the square, and call v_1 the bottom-left vertex, while v_2 is the top-left one; by Property 1, we can have four different situations:

(a) we have a point (p_1) in v_1 and one (p_2) in v_2;

(b) we have a point (p_1) in v_1, and we have 2 other points: one on the left side of the square (p_2), one on the top side (p_3) whose distance is the optimal one m^*;

(c) we have a point (p_2) in v_2, and we have 2 other points: one on the left side of the square (p_1), one on the bottom side (p_4) whose distance is the optimal one m^*;

(d) we have one point on the left side of the square (p_2) and one on the top side (p_3) whose distance is the optimal one m^*; furthermore, we have another point on the left side (p_1) and one on the bottom side (p_4) whose distance is the optimal one m^*.

In all these cases, that are presented in Figure 1, we have at least two points on the left side of the square.

All that remains to be shown is that in cases 1(b), 1(c), and 1(d) the points p_1 and p_2 cannot coincide. For cases (b) and (c) if $p_1 = p_2$ then $m^* > 1$. This is not possible since the optimal distance when $n = 4$ is equal to 1, and for larger instances the distance decreases. Consider now the case (d). Suppose that $p_1 = p_2$, that v_1 has coordinates (0,0), and call y_a the distance between v_1 and p_1 (hence the distance between p_1 and v_2 is equal to $1 - y_a$). The distance between p_1 and p_4 is equal to m^*, and the coordinate x of p_4 is in $(0, 1)$. Similarly, m^* is equal to the distance between p_1 and p_3, and the coordinate x of p_3 is in $(0, 1)$. Hence the following inequalities hold:

$$1 - y_a < m^* < \sqrt{1 + (1 - y_a)^2}$$
$$y_a < m^* < \sqrt{{y_a}^2 + 1},$$

where $y_a \in (0, 1)$. Comparing these inequalities, it turns out that in order to have a valid value of m^* the intervals $\left(1 - y_a, \sqrt{1 + (1 - y_a)^2}\right)$ and

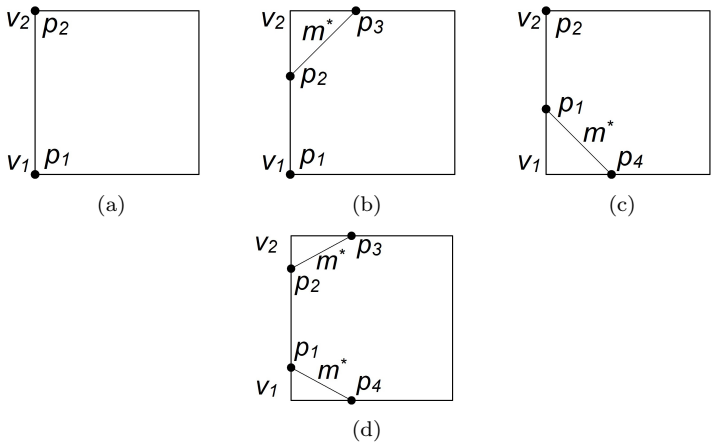

Figure 1: Possible configurations of points in the optimal solution of PPS according to Property 1.

$\left(y_a, \sqrt{{y_a}^2+1}\right)$ must have a nonempty intersection. To ease the following steps we can apply the square operator to the previous inequalities (since all the terms are positive), thus obtaining:

$$(1-y_a)^2 < {m^*}^2 < 1+(1-y_a)^2$$
$${y_a}^2 < {m^*}^2 < {y_a}^2+1.$$

In order to check whether the intersection is nonempty, one should derive an order relationship between the left and right-hand sides of these inequalities. Since $y_a \in (0,1)$, we have actually 3 cases to consider:

- $y_a \in \left(0, \frac{1}{2}\right)$. In this case the order relationship is the following: ${y_a}^2 < (1-y_a)^2 < {y_a}^2+1 < 1+(1-y_a)^2$. Thus, ${m^*}^2$ must be between $(1-y_a)^2$ and ${y_a}^2+1$. In other words, we have that $1-y_a < m^* < \sqrt{{y_a}^2+1}$, $y_a \in \left(0, \frac{1}{2}\right)$;
- $y_a \in \left(\frac{1}{2}, 1\right)$. In this case the order relationship is the following: $(1-y_a)^2 < {y_a}^2 < 1+(1-y_a)^2 < {y_a}^2+1$. Thus, ${m^*}^2$ must be between ${y_a}^2$ and $1+(1-y_a)^2$. In other words, we have that $y_a < m^* < \sqrt{1+(1-y_a)^2}$, $y_a \in \left(\frac{1}{2}, 1\right)$;
- $y_a = \frac{1}{2}$. In this case we have $(1-y_a)^2 = {y_a}^2 = \frac{1}{4}$ and $1+(1-y_a)^2 = {y_a}^2+1 = \frac{5}{4}$. Hence, $\frac{1}{2} < m^* < \frac{\sqrt{5}}{2}$

In the three cases presented above, it is easy to check that m^* must always respect the following constraint:

$$\frac{1}{2} < m^* < \frac{\sqrt{5}}{2}. \tag{10}$$

This means that we can have $p_1 = p_2$ only if the optimal distance satisfies the inequality (10). We are considering the instances having $n \geq 4$. When $n = 4$, the optimal distance is 1, that is less than $\frac{\sqrt{5}}{2}$. The optimal distance when $n = 9$ is equal to $\frac{1}{2}$, and for larger instances the optimal distance decreases. This means that the only instances where p_1 can be equal to p_2 are those where $4 \leq n \leq 8$. However, in these cases the optimal solutions are known, and there are always at least 2 points on a side of the square, as can be checked in [8] or in `http://www.packomania.com`. Hence, it is not possible that p_1 and p_2 coincide.

A similar idea can be used to prove the same for the right side of the square. Moreover, it is true even if we consider the other pair of opposite sides (that is top/bottom) in place of the left/right ones. □

It is actually possible to say more. Using Property 2 and Lemma 1 we can fix $k(n) \geq 2$ points along the left and right sides of the square, where $k(n)$ is monotonically nondecreasing with respect to n. The following theorem formalizes this fact:

Theorem 1. *Consider the PPS problem with $n \geq 4$. There is always an optimal solution where at least $k(n) = 2 + \max\left\{0, \left\lfloor \frac{1}{2U_n} - \frac{\sqrt{2}}{2} \right\rfloor\right\}$ points are on the left side of the square, and at least $k(n)$ points are on the right side of the square.*

Proof. The result of Lemma 1 is that there are at least two points of the optimal solution on the left and right sides of the square. Consider now the left side of the square, and the distance between the points p_1 and p_2 in the four cases depicted in Figure 1; as notation, this distance will be indicated with $d(p_1, p_2)$. To decide if we can put further points on the left side of the square, we need a lower bound on $d(p_1, p_2)$. In the case of Figure 1(a) we have $d(p_1, p_1) = 1$. In the cases of Figures 1(b) and 1(c) we have $d(p_1, p_2) > 1 - m^*$, since p_3 (p_4) can be at distance $\epsilon > 0$ from v_2 (v_1). Finally, for Figure 1(d) one may expect that $d(p_1, p_2) > 1 - 2m^*$. However, in this case (i.e., p_3 and p_4 close respectively to v_2 and v_1) one can take the top and bottom sides of the square to obtain a greater distance between the points playing the role of p_2 and p_1. The worst case is when $d(v_2, p_2) = d(v_2, p_3) = d(v_1, p_1) = d(v_1, p_4) = \frac{m^*}{\sqrt{2}}$, leading to $d(p_1, p_2) \geq 1 - \frac{2m^*}{\sqrt{2}} = 1 - \sqrt{2}m^*$. In these four cases, the lowest value of the lower bound (i.e., the one valid in general) is the last one, that is $d(p_1, p_2) \geq 1 - \sqrt{2}m^*$. Since we do not known m^* (except for $n <= 30$ and $n = 36$), we should use the upper bound to obtain $d(p_1, p_2) \geq 1 - \sqrt{2}m^* \geq 1 - \sqrt{2}U_n$.

Using Property 2 we know that it is not possible to have $d(p_1, p_2) \geq 2U_n$, unless there is another point between p_1 and p_2. Since $d(p_1, p_2)$ is at least $1 - \sqrt{2}U_n$, if $1 - \sqrt{2}U_n \geq 2U_n$ there must be another point between p_1 and p_2. And if $1 - \sqrt{2}U_n \geq 4U_n$ there are at least other 2 points between p_1 and p_2. In general, if $1 - \sqrt{2}U_n \geq 2t(n)U_n$ we have at least $t(n)$ points between p_1 and p_2,

with $t(n) : \mathbb{N}\backslash\{0,1,2,3\} \to \mathbb{Z}$ defined as follows:

$$t(n) = \left\lfloor \frac{1}{2U_n} - \frac{\sqrt{2}}{2} \right\rfloor.$$

Actually, for small instances $t(n)$ could be negative. More precisely, it is the case when $U_n > \frac{\sqrt{2}}{2}$. In order to avoid this issue, we can take the maximum between 0 and $t(n)$. Consequently, the number of points on the left side of the square is at least:

$$k(n) = 2 + \max\{0, t(n)\} = 2 + \max\left\{0, \left\lfloor \frac{1}{2U_n} - \frac{\sqrt{2}}{2} \right\rfloor\right\}. \tag{11}$$

For the right side, the proof is similar. □

Using the result of Theorem 1 we can introduce three corollaries. Before doing that, as the need arises, we should define two sets, representing respectively the indices of the points which are fixed on the left (SL_n) and on the right (SR_n) sides of the square by Theorem 1:

$$SL_n = \{i \in \mathbb{N} : 1 \leq i \leq k(n)\} \tag{12}$$
$$SR_n = \{i \in \mathbb{N} : n - k(n) + 1 \leq i \leq n\}. \tag{13}$$

In other words, Theorem 1 states that :

$$\forall i \in SL_n \quad x_i = 0 \tag{14}$$
$$\forall j \in SR_n \quad x_j = 1. \tag{15}$$

Note that the choice of the points to put in SL_n and SR_n is taken in order to be consistent with the order constraints (26) presented later.

To avoid some symmetries (which can impair performances of Branch-and-Bound based algorithms [11, 12, 13, 14, 15, 16] like COUENNE [7], the solver we use for our tests), we can impose without loss of generality the following conditions:

$$\forall i \in SL_n\backslash\{1\} \quad y_i > y_{i-1} \tag{16}$$
$$\forall j \in SR_n\backslash\{n\} \quad y_{j+1} > y_j. \tag{17}$$

Corollary 1. *Consider the PPS problem with $n \geq 4$. From Theorem 1 we can fix $k(n)$ on the left side of the unit square (i.e., those having indices $i \in SL_n$) and other $k(n)$ points on the right side of the unit square (i.e., those having indices $j \in SR_n$). Then, $\forall i \in SL_n\backslash\{1\}$ $y_i - y_{i-1} < \min\{1, 2U_n\}$ and $\forall j \in SR_n\backslash\{n\}$ $y_{j+1} - y_j < \min\{2U_n, 1\}$.*

Proof. We provide a proof for the left side of the unit square. The proof for the right side is almost the same. From Theorem 1 we have $k(n)$ points on the left side of the square (those having indices $i \in SL_n$). Let m^* be the optimal

solution of the PPS instance considered. From Property 2 and condition (16) we have $\forall i \in SL_n \backslash \{1\}\ y_i - y_{i-1} < 2m^*$. Then, using the upper bound (9), and recalling that the maximum distance between two points along a side of the square is 1, we can write:

$$\forall i \in SL_n \backslash \{1\} \quad y_i - y_{i-1} < 2m^* \leq 2U_n \Rightarrow y_i - y_{i-1} < \min\{1, 2U_n\}. \tag{18}$$

From Property 2 and condition (17) we obtain a similar result for the right side of the square:

$$\forall j \in SR_n \backslash \{n\} \quad y_{j+1} - y_j < 2m^* \leq 2U_n \Rightarrow y_{j+1} - y_j < \min\{1, 2U_n\}. \tag{19}$$

□

It is possible to derive another set of inequalities to tighten the PPS formulation, as proved by the next corollary of Theorem 1.

Corollary 2. *Consider the PPS problem with $n \geq 4$. From Theorem 1 we can fix $k(n)$ points on the left side of the unit square (i.e., those having indices $i \in SL_n$) and other $k(n)$ points on the right side of the unit square (i.e., those having indices $j \in SR_n$). Then, $\forall i \in SL_n \backslash \{1\}\ y_i - y_{i-1} \geq L_n$ and $\forall j \in SR_n \backslash \{n\}\ y_{j+1} - y_j \geq L_n$ hold.*

Proof. Consider the distance constraint (6) involving the points having indices $i \in SL_n$ and $i - 1 \in SL_n : i > 1$ (the same holds for the pairs of points having indices $j \in SR_n$ and $j + 1 \in SR_n : j < n$). At the optimum this constraint can be rewritten as:

$$(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 \geq \alpha^* = m^{*2}.$$

From Theorem 1 it holds that $x_i = x_{i-1}$. Thus, using condition (16) we obtain the following inequalities:

$$\forall i \in SL_n \backslash \{1\} \quad y_i - y_{i-1} \geq m^* \geq L_n \Rightarrow y_i - y_{i-1} \geq L_n. \tag{20}$$

Similarly, for the right side of the square we have:

$$\forall j \in SR_n \backslash \{n\} \quad y_{j+1} - y_j \geq m^* \geq L_n \Rightarrow y_{j+1} - y_j \geq L_n. \tag{21}$$

□

Last corollary is helpful to tighten the range, on the y coordinate, for some of the points which are fixed on the left and right sides of the square.

Corollary 3. *In the optimal solution of PPS, where the points having indices in SL_n are fixed on the left side of the square and the points having indices in*

SR_n are fixed on the right side of the square, the following conditions hold:

$$y_1 \leq \min\left\{U_n, \frac{1}{2}\right\} \tag{22}$$

$$y_{k(n)} \geq \max\left\{1 - U_n, \frac{1}{2}\right\} \tag{23}$$

$$y_{n-k(n)+1} \leq \min\left\{U_n, \frac{1}{2}\right\} \tag{24}$$

$$y_n \geq \max\left\{1 - U_n, \frac{1}{2}\right\} \tag{25}$$

Proof. Consider the cases depicted in Figure 1. Intuitively, the corollary states that among the $k(n)$ points fixed on the left (right) side of the square, one is on the bottom-half, and one in on the top-half. According to Theorem 1, and to (16)-(17), these points are respectively those having indices 1 and $k(n)$ ($n - k(n) + 1$ and n for the right side of the square).

Consider the left side of the square. In the worst case (in terms of distance of a point from the closest vertex) the distance between the bottom-left vertex and the point with index 1 is smaller than m^*, that is smaller than or equal to U_n, and the same happens for the distance between the point with index $k(n)$ and the top-left vertex. Moreover, we can write without loss of generality that $y_1 \leq \frac{1}{2}$ and $y_{k(n)} \geq \frac{1}{2}$. Hence, we can conclude that (22) and (23) are valid. In a similar way we can prove the result for the right side of the square. □

Using Theorem 1 and Corollaries 1, 2, and 3 we can improve the PPS formulation (1)-(5), by adjoining constraints (14), (15), (18), (19), (20), (21), (22), (23), (24), and (25). However, some considerations lead to further constraints which can be used to tighten the formulation. They are presented in the next section.

2.2. Other constraints

A first set of inequalities which can be adjoined to the PPS formulation are the order symmetry-breaking constraints on the x variables [11, 12, 16]:

$$\forall i \in N \backslash \{n\} \quad x_i \leq x_{i+1}. \tag{26}$$

However, using (14)-(15) we can reduce the number of such constraints:

$$\forall i \in N \backslash \{SL_n \cup SR_n \cup \{n - k(n)\}\} \quad x_i \leq x_{i+1}. \tag{27}$$

Another set of symmetry-breaking constraints can be derived from the following facts [17]:

- at least $n_x = \lceil \frac{n}{2} \rceil$ points are on the left-half of the square (we call it x *bounds constraints*);
- among the previous n_x points, at least $n_y = \lceil \frac{n_x}{2} \rceil$ are on the bottom-half (y *bounds constraints*).

Note that these constraints also hold if n_x points are placed on the right-half of the square, and among them n_y are placed on the top-half of the square (this is actually how they are presented in [17]). Unfortunately, in general we cannot have the *x bounds constraints*, the *y bounds constraints* and the order constraints (27) together. It is possible to do it by removing from (27) the inequality $x_{n_y} \leq x_{n_y+1}$. However, we need to preserve the order constraints to derive the *triangular inequality constraints* (31) presented later. Moreover, due to constraints (16)-(17), the *y bounds constraints* are difficult to express. Hence, we only consider the *x bounds constraints* to improve the PPS formulation. Using (27), these constraints can be expressed by means of a single inequality:

$$x_{n_x} \leq \frac{1}{2}. \tag{28}$$

In the general case, another constraint which can be adjoined is the following:

$$y_2 - y_1 \leq y_{n-k(n)+2} - y_{n-k(n)+1}. \tag{29}$$

However, due to constraints (27) and (28), the constraint (29) could not hold. As a matter of fact, if n is an odd number we are imposing that more than half on the points are on the left side of the square, thus we cannot a priori say if (29) is valid or not. Hence, we adjoin (29) to our model only if n is an even number. Moreover, in case that $k(n) = 2$ and n is an even number, we can combine (29) with (18), (19), (20), and (21) to obtain a stronger condition:

$$L_n \leq y_2 - y_1 \leq y_n - y_{n-1} < \min\{1, 2U_n\}. \tag{30}$$

To resume, in our improved model we have the constraints (18), (19), (20), and (21). If n is an even number and $k(n) = 2$, the constraints (18)-(21) are replaced by (30). If n is an even number but $k(n) > 2$, then (29) is adjoined to the PPS formulation. Otherwise the model is not changed.

A final set of inequalities can be derived from the triangular inequality between pairs points. The triangular inequalities can be written as:

$$\forall i \in N, \forall j \in N \,:\, i < j \quad |x_j - x_i| + |y_j - y_i| \geq d(i,j) \geq m = \sqrt{\alpha}$$

where $d(i,j)$ represents the distance between the points having indices i and j. First, we can remove the absolute value from the x variable, since the order constraints (27) hold. Moreover, we can avoid to define these constraints for the pairs of indices (i,j) belonging to SL_n or SR_n, because in this case $x_j - x_i = 0$ and the resulting constraint is implied by (6). Hence, the triangular inequalities are defined for pairs of indices belonging to $T = \{(i,j) \,:\, i \in N, j \in N, i < j, \neg((i \in SL_n \wedge j \in SL_n) \vee (i \in SR_n \wedge j \in SR_n))\}$. Now we can write:

$$\forall (i,j) \in T \quad x_j - x_i + |y_j - y_i| \geq \sqrt{\alpha}.$$

We are interested in finding linear inequalities, because we do not want to worsen the formulation with additional nonlinear constraints. To do this, we should

remove the absolute value and the square root. The former can be done by relaxing the constraint by means of the following inequality: $y_j + y_i \geq |y_j - y_i|$. The latter can be done by noticing that the optimal distance for the instances where $n \geq 4$ is in $(0, 1]$. This means that $\sqrt{\alpha} \geq \alpha$. Actually we can obtain a tighter bound, i.e., we want to find the maximum value for a parameter $p \in \mathbb{R} : p \geq 1$ such that $\sqrt{\alpha} \geq p\alpha$. Hence, we have:

$$\sqrt{\alpha} \geq p\alpha \Rightarrow p^2 \leq \frac{1}{\alpha}.$$

Since we are interested in the maximum value of p, we can write:

$$p^2 = \frac{1}{\alpha} \Rightarrow p = \frac{1}{\sqrt{\alpha}} \geq \frac{1}{U_n}.$$

Since p must be greater than or equal to 1, it can be determined as:

$$p = \max\left\{1, \frac{1}{U_n}\right\}.$$

Finally, we have:

$$\forall (i,j) \in T \quad x_j - x_i + y_j + y_i \geq x_j - x_i + |y_j - y_i| \geq \sqrt{\alpha} \geq p\alpha,$$

leading to the final form of the triangular inequality constraints which will be adjoined to our PPS model:

$$\forall (i,j) \in T \quad x_j - x_i + y_j + y_i \geq \max\left\{1, \frac{1}{U_n}\right\} \alpha. \tag{31}$$

Note that the only case, among those considered, for which $\frac{1}{U_n} < 1$ (using the upper bound definition in (9)) is when $n = 4$. However, for the sake of completeness we provide a constraint that is valid for the general case.

2.3. Improved mathematical programming formulation

Using the results of Theorem 1, Corollary 1, Corollary 2, Corollary 3 and the inequalities presented in the previous section, we can modify the PPS formulation (1)-(5). As notation, $\mathrm{mod}\,(n, 2)$ represents the modulo operation, and it is equal to 0 if n is an even number, 1 otherwise. Moreover, let τ_n be equal

to $\min\{1, 2U_n\}$. The improved PPS formulation can be defined as:

$$\max \quad \alpha \tag{32}$$

$$\text{s.t.} \quad \forall i \in N, \forall j \in N : i < j \quad x_i^2 + x_j^2 - 2x_i x_j + y_i^2 + y_j^2 - 2y_i y_j \geq \alpha \tag{33}$$

$$\forall i \in N \backslash \{SL_n \cup SR_n\} \quad x_i \in [0, 1] \tag{34}$$

$$\forall i \in N \quad y_i \in [0, 1] \tag{35}$$

$$\forall i \in SL_n \quad x_i = 0 \tag{36}$$

$$\forall j \in SR_n \quad x_j = 1 \tag{37}$$

$$\text{if } (\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad y_2 - y_1 \geq L_n \tag{38}$$

$$\text{if } (\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad y_2 - y_1 \leq y_n - y_{n-1} \tag{39}$$

$$\text{if } (\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad y_n - y_{n-1} \leq \tau_n \tag{40}$$

$$\text{if } (\text{mod}(n, 2) = 0 \wedge k(n) > 2) \quad y_2 - y_1 \leq y_{n-k(n)+2} - y_{n-k(n)+1} \tag{41}$$

$$\text{if } \neg(\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad \forall i \in SL_n \backslash \{1\} \quad y_i - y_{i-1} \leq \tau_n \tag{42}$$

$$\text{if } \neg(\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad \forall j \in SR_n \backslash \{n\} \quad y_{j+1} - y_j \leq \tau_n \tag{43}$$

$$\text{if } \neg(\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad \forall i \in SL_n \backslash \{1\} \quad y_i - y_{i-1} \geq L_n \tag{44}$$

$$\text{if } \neg(\text{mod}(n, 2) = 0 \wedge k(n) = 2) \quad \forall j \in SR_n \backslash \{n\} \quad y_{j+1} - y_j \geq L_n \tag{45}$$

$$y_1 \leq \frac{\tau_n}{2} \tag{46}$$

$$y_{k(n)} \geq \max\left\{1 - U_n, \frac{1}{2}\right\} \tag{47}$$

$$y_{n-k(n)+1} \leq \frac{\tau_n}{2} \tag{48}$$

$$y_n \geq \max\left\{1 - U_n, \frac{1}{2}\right\} \tag{49}$$

$$\forall i \in N \backslash \{SL_n \cup SR_n \cup \{n - k(n)\}\} \quad x_i \leq x_{i+1} \tag{50}$$

$$x_{n_x} \leq \frac{1}{2} \tag{51}$$

$$\forall (i, j) \in T \quad x_j - x_i + y_j + y_i \geq \max\left\{1, \frac{1}{U_n}\right\} \alpha \tag{52}$$

$$\alpha \in \mathbb{R}. \tag{53}$$

Note that we replaced the strict inequality of (18), (19), and (30) with nonstrict ones in (42), (43), and (40), because it is much easier to deal with nonstrict inequalities (otherwise the feasible region could be an open set). However, this does not change the global optimum: a solution which makes the left-hand side of one of those inequalities equal to its right-hand side would not be the optimal according to Property 2, unless in the optimal solution there is another point between those involved in this inequality.

3. Computational results

In order to show the effect of these constraints, we now compare the results obtained by solving some PPS instances, using the original formulation (1)-(5)

and the improved formulation (32)-(53). The results have been obtained on a 2.4 GHz Intel Xeon CPU computer with 24 GB RAM running Linux and the solver COUENNE 0.4, which implements a spatial Branch-and-Bound (sBB) algorithm (the sBB algorithm is an ϵ-approximation algorithm for solving non-convex Nonlinear Programming (NLPs) problems and Mixed Integer Nonlinear Programming (MINLPs) problems [18]). We report the value of the best known distance m^*, found on `www.packomania.com` (if in bold, this means that it has been proved to be the optimal distance). Then, for each formulation, we report the best value of distance ($\sqrt{LB}$), the best value of α (LB) and the upper bound on α (UB) found by COUENNE within 2 hours, the gap still open (we use the CPLEX definition [19]: $\left(\frac{100 \cdot |LB-UB|}{|LB|+10^{-10}}\right)$ %), the number of sBB nodes explored, and the CPU time in seconds, with a time limit of 2 hours (the symbol $*$ indicates that the time limit is reached). Note that a gap of 0% means that the optimal solution has been found. Values in bold are the best among the two formulations; all the values related to the distance (i.e., m^*, $\sqrt{LB}$, LB and UB) are rounded to 4 decimal places.

4. Conclusion

In this paper we study the PPS problem from both the theoretical and computational points of view. Starting from some properties presented in [9, 10], we derive a theorem and some corollaries. Using these results, together with some other properties presented in the second part of the Section 2.2 and some symmetry breaking constraints proposed in [11, 16], we derive an improved formulation for the PPS problem.

The computational experiments performed on some instances of PPS show that the improved formulation is more efficient for small instances (solved within the time limit), as the optimal solution is found in less time. For larger instances, the improved formulation gives better values of the upper bound; one of the reason for this behavior is that the improved formulation changes the bounds for some of the variables, and this has been shown to have a high impact on the solution process of the PPS problem in [11, 15]. This difference in the upper bound value is also underlined by the gap, which is much lower for the improved formulation, and also seems to increase more slowly with respect to the original formulation.

Moreover, by comparing the number of sBB nodes, it appears that when it is possible to solve the problem within the time limit, the sBB tree is smaller for the improved formulation. If the time limit is reached, the number of sBB nodes explored is lower for the original formulation, meaning that each subproblem is more difficult to solve.

A very interesting fact that can be noted is that the original formulation provides very good values of the lower bound. In other words, using the original formulation it is possible to find solutions which are the same, or very close, to the best known solutions so far. As future work, this idea could be used to devise a Branch-and-Bound based algorithm to solve the PPS problem based on the

		Original formulation						Improved formulation					
n	m^*	$\sqrt{LB}$	LB	UB	gap %	sBB nodes	CPU time	$\sqrt{LB}$	LB	UB	gap %	sBB nodes	CPU time
4	**1**	1	1	1	0	90	0.94	1	1	1	0	**0**	**0.03**
5	**0.7071**	0.7071	0.5	0.5	0	6,667	9.06	0.7071	0.5	0.5	0	**0**	**0.18**
6	**0.6009**	0.6009	0.3611	0.3611	0	963,499	1421.49	0.6009	0.3611	0.3611	0	164	**1.46**
7	**0.5359**	0.5359	0.2872	0.4021	40.01	2,906,775	7,200*	0.5359	0.2872	**0.2872**	**0**	1,535	**5.03**
8	**0.5176**	0.5176	0.2679	0.4941	84.43	2,098,921	7,200*	0.5176	0.2679	**0.2679**	**0**	6,376	**20.11**
9	**0.5**	0.5	0.25	0.5	100.00	1,389,918	7,200*	0.5	0.25	**0.25**	**0**	18,458	**63.53**
10	**0.4213**	0.4213	0.1775	0.5036	183.72	1,353,701	7,200*	0.4213	0.1775	**0.1775**	**0**	1,438,562	**4791.66**
11	**0.3982**	**0.3982**	**0.1586**	0.5077	220.11	825,030	7,200*	0.3980	0.1584	**0.1940**	**22.47**	1,682,724	7,200*
12	**0.3887**	0.3887	0.1511	0.5109	238.12	623,710	7,200*	0.3887	0.1511	**0.2281**	**50.96**	1,040,836	7,200*
13	**0.3661**	**0.3661**	**0.1340**	0.8331	521.72	489,671	7,200*	0.3555	0.1264	**0.2521**	**99.45**	833,256	7,200*
14	**0.3489**	**0.3489**	**0.1217**	0.9530	683.07	407,215	7,200*	0.3480	0.1211	**0.2815**	**132.45**	677,372	7,200*
15	**0.3411**	**0.3411**	**0.1163**	0.9604	725.80	338,281	7,200*	0.3339	0.1115	**0.3081**	**176.32**	577,482	7,200*
16	**0.3333**	**0.3333**	**0.1111**	0.9685	771.74	284,220	7,200*	0.3129	0.0979	**0.3066**	**213.18**	481,512	7,200*
17	**0.3062**	**0.3062**	**0.0937**	0.9697	964.03	245,477	7,200*	0.2994	0.0896	**0.2995**	**234.26**	406,807	7,200*
18	**0.3005**	**0.3005**	**0.0903**	0.9771	982.06	207,707	7,200*	0.2987	0.0892	**0.2889**	**223.88**	440,880	7,200*
19	**0.2895**	**0.2895**	**0.0838**	0.9756	1064.20	179,246	7,200*	0.2782	0.0774	**0.2814**	**263.57**	297,878	7,200*
20	**0.2866**	**0.2866**	**0.0821**	0.9787	1092.08	153,955	7,200*	0.2795	0.0781	**0.2761**	**253.52**	335,925	7,200*
21	**0.2718**	**0.2718**	**0.0739**	0.9804	1226.66	137,755	7,200*	0.2681	0.0719	**0.2795**	**288.73**	196,908	7,200*
22	**0.2680**	**0.2680**	**0.0718**	0.9773	1261.14	118,984	7,200*	0.2567	0.0659	**0.2706**	**310.62**	227,138	7,200*
23	**0.2588**	**0.2587**	**0.0669**	1.25	1768.46	108,941	7,200*	0.2504	0.0627	**0.2686**	**328.39**	194,857	7,200*
24	**0.2543**	**0.2543**	**0.0647**	0.9788	1412.83	92,997	7,200*	0.25	0.0625	**0.2783**	**345.28**	165,955	7,200*
25	**0.25**	**0.25**	**0.0625**	0.9778	1464.48	82,985	7,200*	0.2360	0.0557	**0.2748**	**393.36**	144,203	7,200*
26	**0.2387**	**0.2387**	**0.0570**	0.9790	1617.54	76,460	7,200*	0.2265	0.0513	**0.2713**	**428.85**	122,228	7,200*
27	**0.2358**	**0.2358**	**0.0556**	1.25	2148.20	67,465	7,200*	0.2229	0.0497	**0.2664**	**436.02**	114,467	7,200*
28	**0.2305**	**0.2305**	**0.0532**	0.9773	1737.03	59,132	7,200*	0.2117	0.0448	**0.2623**	**485.49**	105,330	7,200*
29	**0.2269**	**0.2269**	**0.0515**	0.9780	1799.03	51,351	7,200*	0.2142	0.0459	**0.2577**	**461.44**	88,318	7,200*
30	**0.2245**	**0.2245**	**0.0504**	0.9850	1854.37	46,519	7,200*	0.2131	0.0454	**0.2555**	**462.78**	75,783	7,200*
31	0.2175	**0.2175**	**0.0473**	0.9939	2000.13	41,968	7,200*	0.2058	0.0424	**0.2498**	**489.15**	64,332	7,200*
32	0.2132	**0.2132**	**0.0454**	1.25	2653.30	40,704	7,200*	0.1992	0.0397	**0.2475**	**523.43**	60,377	7,200*
33	0.2113	**0.2113**	**0.0447**	1.0013	2140.04	34,724	7,200*	0.1945	0.0378	**0.2422**	**540.75**	55,083	7,200*
34	0.2056	**0.2052**	**0.0421**	1.25	2869.12	33,947	7,200*	0.1886	0.0356	**0.2418**	**579.21**	49,583	7,200*
35	0.2028	**0.1986**	**0.0395**	1.25	3064.56	29,648	7,200*	0.1859	0.0346	**0.2374**	**586.13**	40,866	7,200*
36	**0.2**	**0.1954**	**0.0382**	1.25	3172.25	26,752	7,200*	0.1887	0.0356	**0.2303**	**546.91**	34,384	7,200*
37	0.1964	**0.1946**	**0.0379**	1.0252	2605.01	21,345	7,200*	0.1853	0.0343	**0.2221**	**547.52**	38,653	7,200*
38	0.1953	0.1723	0.0297	1.0429	3411.48	22,956	7,200*	**0.1830**	**0.0335**	**0.2142**	**539.40**	33,760	7,200*
39	0.1944	**0.1858**	**0.0345**	1.25	3523.19	20,482	7,200*	0.1813	0.0329	**0.2065**	**527.66**	28,849	7,200*
40	0.1882	**0.1878**	**0.0353**	1.0346	2830.89	16,681	7,200*	0.1655	0.0274	**0.1984**	**624.09**	42,787	7,200*

Table 1: Performances of the COUENNE solver on small PPS instances.

formulations proposed in this paper: one could employ the original formulation to find good solutions, and then use the improved formulation to obtain good upper bound values that can be used to cut nodes from the Branch-and-Bound tree. This idea could be tested, using also other solvers as BARON [20], to prove the optimality for some of the smallest open instances (e.g., 31, 32, 33) for which the solution found within two hours using the original formulation is almost as good as the best known solution.

Acknowledgements

The author would like to thank the anonymous referees for their suggestions and comments. Moreover, the author thanks Pietro Belotti and Giacomo Nannicini for the interesting discussions about the solver Couenne. This research was supported by IDC Grant IDSF1200101OH.

References

[1] M. Hifi, V. Th. Paschos, V. Zissimopoulos, A simulated annealing approach for the circular cutting problem, European Journal of Operational Research 159 (2) (2004) 430 – 448.

[2] M. Hifi, R. M'Hallah, Approximate algorithms for constrained circular cutting problems, Computers & Operations Research 31 (5) (2004) 675 – 694.

[3] Y. Cui, Generating optimal t-shape cutting patterns for circular blanks, Computers & Operations Research 32 (1) (2005) 143 – 152.

[4] H. J. Fraser, J. A. George, Integrated container loading software for pulp and paper industry, European Journal of Operational Research 77 (3) (1994) 466 – 474.

[5] J. A. George, J. M. George, B. W. Lamar, Packing different-sized circles into a rectangular container, European Journal of Operational Research 84 (3) (1995) 693 – 712.

[6] I. Castillo, F. Kampas, J. Pintér, Solving circle packing problems by global optimization: Numerical results and industrial applications, European Journal of Operational Research 191 (3) (2008) 786 – 802.

[7] P. Belotti, J. Lee, L. Liberti, F. Margot, A. Wächter, Branching and bounds tightening techniques for non-convex MINLP, Optimization Methods and Software 24 (4) (2009) 597 – 634.

[8] P. G. Szabó, M. C. Markót, T. Csendes, E. Specht, L. G. Casado, I. García, New Approaches to Circle Packing in a Square: With Program Codes (Springer Optimization and Its Applications), Springer-Verlag New York, 2007.

[9] M. Locatelli, U. Raber, Packing Equal Circles in a Square: I. Theoretical Results, Tech. Rep. 08-99, Dipartimento Sistemi e Informatica, Università di Firenze (1999).

[10] M. Locatelli, U. Raber, Packing equal circles in a square: a deterministic global optimization approach, Discrete Applied Mathematics 122 (1-3) (2002) 139 – 166.

[11] A. Costa, Applications of Reformulations in Mathematical Programming, Ph.D. thesis, École Polytechnique, France (2012).

[12] A. Costa, I. Tseveendorj, Symmetry breaking constraints for the problem of packing equal circles in a square, in: C. J. Luz, F. Valente (Eds.), Proceedings of the 1^{st} International Conference on Operations Research and Enterprise Systems (ICORES), SciTePress, 2012, pp. 5 – 10.

[13] L. Liberti, Symmetry in mathematical programming, in: J. Lee, S. Leyffer (Eds.), Mixed Integer Nonlinear Programming, Vol. 154 of IMA, Springer, New York, 2012, pp. 263 – 286.

[14] L. Liberti, Reformulations in mathematical programming: automatic symmetry detection and exploitation, Mathematical Programming 131 (2012) 273 – 304.

[15] A. Costa, P. Hansen, L. Liberti, Bound constraints for Point Packing in a Square, in: L. Adacher, M. Flamini, G. Leo, G. Nicosia, A. Pacifici, V. Piccialli (Eds.), Proceedings of the 10^{th} Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW), Università di Roma "Tor Vergata", Villa Mondragone, 2011, pp. 126 – 129.

[16] A. Costa, P. Hansen, L. Liberti, On the impact of symmetry-breaking constraints on spatial Branch-and-Bound for circle packing in a square, Discrete Applied Mathematics 161 (1-2) (2013) 96 – 106.

[17] K. Anstreicher, Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming, Journal of Global Optimization 43 (2) (2009) 471 – 484.

[18] L. Liberti, Introduction to Global Optimization, Tech. rep., LIX, École Polytechnique (2008).

[19] IBM, ILOG CPLEX 12.2 User's Manual, IBM (2010).

[20] N. V. Sahinidis, BARON 12.1.0: Global Optimization of Mixed-Integer Nonlinear Programs, *User's Manual* (2013).