Submitted to *INFORMS Journal on Computing*
manuscript JOC-2022-01-OA-004, accepted 31 January 2023

Perturbation-based thresholding search for packing equal circles and spheres

Xiangjing Lai
Institute of Advanced Technology, Nanjing University of Posts and Telecommunications, Nanjing 210023, China, laixiangjing@gmail.com

Jin-Kao Hao* (Corresponding author)
LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers, France, jin-kao.hao@univ-angers.fr

Renbin Xiao
School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China, rbxiao@hust.edu.cn

Fred Glover
ECEE - College of Engineering & Applied Science, University of Colorado, Boulder, Colorado 80309, USA, fredwglover@yahoo.com

This paper presents an effective perturbation-based thresholding search for two popular and challenging packing problems with minimal containers: packing N identical circles in a square and packing N identical spheres in a cube. Following the penalty function approach, we handle these constrained optimization problems by solving a series of unconstrained optimization subproblems with fixed containers. The proposed algorithm relies on a two-phase search strategy that combines a thresholding search method reinforced by two general-purpose perturbation operators and a container adjustment method. The performance of the algorithm is assessed relative to a large number of benchmark instances widely-studied in the literature. Computational results show a high performance of the algorithm on both problems compared to the state-of-the-art results. For circle packing, the algorithm improves 156 best-known results (new upper bounds) in the range of $2 \leq N \leq 400$ and matches 242 other best-known results. For sphere packing, the algorithm improves 66 best-known results in the range of $2 \leq N \leq 200$, while matching the best-known results for 124 other instances. Experimental analyses are conducted to shed light on the main search ingredients of the proposed algorithm consisting of the two-phase search strategy, the mixed perturbation and the parameters.

Key words: Circle and sphere packing, global optimization, constrained optimization, nonlinear non-convex optimization, heuristics.

1. Introduction

Packing N non-overlapping objects (e.g., circles, spheres) in a regular convex container (e.g., square, cube in Euclidean space R^d, $d \geq 2$) such that the packing density is maximized covers a class of widely studied geometrical optimization problems (Addis et al. (2008), Weaire and Aste (2008)). For the two-dimensional case with $d = 2$, the corresponding problems are called circle packing problems, among which the problem of Packing Equal Circles in a Square (PECS) is the most representative (Addis et al. (2008), Schaer (1965)). For the three-dimensional case with $d = 3$, the corresponding problems are called sphere packing with the problem of Packing Equal Spheres in a Cube (PESC) (Schaer (1966)) being the most studied.

Given N unit circles $\{c_1, c_2, \ldots, c_N\}$, PECS consists of packing these N circles into a square container without overlap, such that the size of the square container is minimized. Formally, PECS can be expressed as a nonlinear programming problem as follows:

$$\text{Minimize} \quad L \tag{1}$$

$$\text{Subject to} \quad \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq 2, 1 \leq i \neq j \leq N \tag{2}$$

$$|x_i| + 1.0 \leq \frac{1}{2}L, i = 1, 2, \ldots, N \tag{3}$$

$$|y_i| + 1.0 \leq \frac{1}{2}L, i = 1, 2, \ldots, N \tag{4}$$

where L is the size of the square container (i.e., the length of sides) with a center at the origin of the two-dimensional Cartesian coordinate system and (x_i, y_i) and (x_j, y_j) denote respectively the centers of unit circles c_i and c_j. The constraints (2) are called the non-overlapping constraints and guarantee that no overlap occurs between any two unit circles, and the constraints (3) and (4) are called the containment constraints and ensure that all N unit circles are completely contained in the square container. The formulation can be easily extended to PESC in R^3 space as follows:

$$\text{Minimize} \quad L \tag{5}$$

$$\text{Subject to} \quad \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \geq 2, 1 \leq i \neq j \leq N \tag{6}$$

$$|x_i| + 1.0 \leq \frac{1}{2}L, i = 1, 2, \ldots, N \tag{7}$$

$$|y_i| + 1.0 \leq \frac{1}{2}L, i = 1, 2, \ldots, N \tag{8}$$

$$|z_i| + 1.0 \leq \frac{1}{2}L, i = 1, 2, \ldots, N \tag{9}$$

where (x_i, y_i, z_i) and (x_j, y_j, z_j) denote respectively the centers of unit spheres s_i and s_j, and L is the size of cube container. The constraints (6) ensure that any two unit spheres do not overlap, and the constraints (7)–(9) ensure that all N unit spheres are contained in the cube container.

PECS is the most widely studied circle packing problem and is also known as the point arrangement problem in a square (Akiyama et al. (2002), Casado et al. (1998), Costa (2013), Goldberg (1970)) and as the continuous N-dispersion problem in a square (Dai et al. (2021), Dimnaku et al. (2005), Drezner and Erkut (1995)). PECS can also be viewed as a special case of Latin hypercube design (LHD) with Euclidean distance in a square (Van Dam et al. (2007)). In addition, an equivalent formulation of PECS is that of packing N congruent circles into a unit square such that the common radius r of circles is maximized (Addis et al. (2008)).

Both PECS and PESC are general and natural models that can formulate a number of applications, including circular cutting, container loading, facility layout, and so on (Castillo et al. (2008)). Thus, developing high-performance algorithms for PECS and PESC can be useful for solving these real-world applications. It has been established that PECS and PESC are NP-hard (Demaine et al. (2010)), making it unlikely to solve these problems exactly. As a result, it is of value to develop effective heuristic algorithms able to find high-quality solutions for challenging problem instances.

This work is dedicated to the design and implementation of a high-performance heuristic algorithm for both PECS and PESC, with the following contributions. First, a perturbation-based thresholding search (PBTS) algorithm is proposed to solve the constrained PECS and PESC problems by solving a series of unconstrained subproblems with a penalty function approach. For each unconstrained subproblem with a fixed container size L, PBTS employs a thresholding search procedure and a container adjustment procedure to explore various candidate configurations. The thresholding search integrates two types of perturbations and an effective local optimization method. Second, we provide extensive computational results on popular benchmark instances of PECS and PESC and report 156 new best-known results for PECS in the range of $2 \leq N \leq 400$ and 66 new best-known results for PESC in the range of $2 \leq N \leq 200$. Third, we investigate the main components of the proposed algorithm to shed light on their roles.

The rest of paper is organized as follows. In Section 2, we review the related studies in the literature. In Section 3, we give an unconstrained optimization formulation for the PECS and PESC problems with a container of fixed size. Section 4 describes the proposed algorithm in detail. The algorithm's performance is assessed by extensive computational experiments in Section 5 and several essential components of the algorithm are analyzed and discussed in Section 6. The concluding section summarizes our findings and provides research perspectives for future research.

2. Literature Review

Since the pioneering study of Schaer (1965), PECS and PESC have become the subject of widespread research, leading to a large number of theoretical and computational results. For a comprehensive review of the PECS studies prior to 2006, the reader is referred to the book of Szabó et al. (2007). Below we briefly review the most representative studies of PECS and PESC, including both exact methods and heuristic methods.

2.1. Exact methods

Table 1 Exact approaches for PECS and PESC.

Year	Approach	Authors	N	Problem
1965	mathematical method	Schaer (1965)	$N \leq 9$	PECS
1966	mathematical method	Schaer (1966)	$N \leq 9$	PESC
1970	mathematical method	Schwartz (1970)	$N = 6$	PESC
1983	mathematical method	Wengerodt (1983)	$N = 16$	PESC
1987	mathematical method	Wengerodt (1987)	$N = 14$	PESC
1987	mathematical method	Wengerodt and Kirchner (1987)	$N = 36$	PESC
1999	mathematical method	Hujter (1999)	$N = 10$	PECS
1999	computer-aided approach	Nurmela et al. (1999)	$N \leq 27$	PECS
2002	computer-aided approach	Locatelli and Raber (2002)	$\{10-35, 38, 39\}$	PECS
2005	computer-aided approach	Markót and Csendes (2005)	$N \leq 30$	PECS
2009	mathematical method	Joós (2009)	$N = 14$	PESC
2013	computer-aided approach	Costa et al. (2013)	$N \leq 50$	PECS
2013	computer-aided approach	Costa (2013)	$N \leq 40$	PECS
2021	computer-aided approach	Markót (2021)	$N \leq 33$	PECS

Table 1 summarizes the main studies related to exact methods, which focus on obtaining a proven optimal solution. These studies adopt two approaches. The first approach aims to prove global optimality of a solution by means of a mathematical method, as in the work of Schaer (1965, 1966), Schwartz (1970), Wengerodt (1983, 1987), Wengerodt and Kirchner (1987), Hujter (1999), Joós (2009). For example, Schaer (1965, 1966) provided a proof of optimality for solutions to some small PECS and PESC instances with $N \leq 9$. In 2009, Joós (2009) provided an optimality proof for the PESC solution with $N = 14$.

The second approach aims to determine the optimal solution by a computer-aided method. For example, Nurmela et al. (1999) determined the optimal solutions of PESC for $N \leq 27$ using a specific method to enumerate the possible combinations. In 2002, Locatelli and Raber (2002) found the optimal PECS solutions within a precision of 10^{-5} for $N \leq 35$, 38 and 39 by a branch-and-bound algorithm with several efficient pruning strategies. In 2005, Markót and Csendes (2005) determined the optimal PECS solutions for the instances with up to $N = 30$ circles by a branch-and-bound algorithm based on interval analysis, and Markót (2021) recently improved the algorithm and extended the results to $N = 33$. Costa et al. (2013) proposed an improved branch-and-bound algorithm by considering the impact of symmetry-breaking constraints.

2.2. Heuristic methods

Table 2 Heuristic approaches for PECS and PESC.

Year	Approach	Authors	N	Problem
1971	constructive method	Goldberg (1971)	$N \leq 27$	PESC
1995	multi-start method	Maranas et al. (1995)	$N \leq 30$	PECS
1996	billiard simulation method	Graham and Lubachevsky (1996)	$N \leq 61$	PECS
1997	multi-start method	Nurmela and Östergård (1997)	$N \leq 50$	PECS
1998	TAMSASS-PECS	Casado et al. (1998)	$N \leq 100$	PECS
2000	two-phase method	Boll et al. (2000)	$N \leq 50$	PECS
2004	perturbed billiard simulation	Gensane (2004)	$N \leq 32$	PESC
2007	modified billiard simulation	Szabó and Specht (2007)	$N \leq 200$	PECS
2008	monotonic basin hopping	Addis et al. (2008)	$N \leq 130$	PECS , PESC
2008	population basin hopping	Addis et al. (2008)	$N \leq 130$	PECS , PESC
2010	greedy vacancy search	Huang and Ye (2010)	$N \leq 200$	PECS
2012	variable neighborhood search	M'Hallah and Alkandari (2012)	$N \leq 55$	PESC
2019	clustering-based heuristic	Bagattini et al. (2019)	$N \leq 120$	PESC
2022	two-phase heuristic	Amore and Morales (2022)	$254 \leq N \leq 9996$	PECS

Heuristic methods undertake to find a high-quality (not necessarily optimal) solution within a reasonable computation time. The related PECS and PESC studies are summarized in Table 2, including the most representative approaches such as multi-start algorithms (Maranas et al. (1995), Nurmela and Östergård (1997)), billiard simulation methods and their variants (Graham and Lubachevsky (1996), Gensane (2004), Szabó and Specht (2007)), threshold accepting search (Casado et al. (1998)), monotonic basin hopping search and its variants (Addis et al. (2008)), the clustering-based heuristic (Bagattini et al. (2019)), variable neighborhood search (M'Hallah and Alkandari (2012)), greedy vacancy search (Huang and Ye (2010)), and two-phase optimization (Amore and Morales (2022), Boll et al. (2000)).

In detail, Maranas et al. (1995) proposed a multi-start algorithm for PECS and reported computational results for $n \leq 30$, including an improved solution for $N = 15$. The algorithm employs a randomized initialization method and a non-linear programming solver as its local optimization procedure. In 1996, using a billiard simulation method, Graham and Lubachevsky (1996) reported several dense packing configurations for PECS and investigated repeated patterns for dense packing configurations. In 1997, Nurmela and Östergård (1997) reported the putative optimum solutions for $N \leq 50$ with a multi-start strategy to minimize the energy function $E(X) = \sum_{i<j}(\frac{\lambda}{d_{ij}^2})^M$ where d_{ij} is the distance between the centers of two circles c_i and c_j that are restricted to lie in a unit square, λ is a scaling factor, and M is a positive integer whose value defines an energy function. Starting with a small M value (in the range $10 \leq M \leq 100$) and a random initial solution, the algorithm minimizes a sequence of energy functions $E(X)$ with an increasing M value. When M reaches a very large value (e.g., $M = 10^6$), the solutions of $E(X)$ converge to a local optimal solution of PECS (Hardin and Saff (2005)), where the minimum distance between the centers of circles is the common diameter of circles. In 1998, Casado et al. (1998) proposed a threshold accepting algorithm for PECS, called TAMSASS-PECS, which gradually decreases the threshold value used to accept the new solutions as the search progresses, like the temperature parameter in simulated annealing. The algorithm was evaluated on the instances with $N \leq 100$.

Several important studies have been reported during the next ten years. In 2000, Boll et al. (2000) proposed a two-phase algorithm for PECS by combining two heuristic strategies, and reported improved packing configurations for $N = 32, 37, 48, 50$. In 2004, Gensane (2004) proposed a perturbed billiard simulation method for PESC and improved the best-known results for all instances in the range of $11 \leq N \leq 26$ except for $N = 13, 14, 18$. In 2007, Szabó and Specht (2007) obtained putatively optimal configurations by means of a modified billiard simulation algorithm for PECS instances with $N \leq 200$. Based on the conjecture that the objective functions of PECS and PECS have a funneling landscape, Addis et al. (2008) proposed the monotonic basin hopping algorithm and the population basin hopping algorithm. These algorithms improved the previous best-known results for 32 PECS instances in the range of $N \leq 130$ and one PESC instance with $N = 28$.

Considering the research conducted since 2010, the following studies deserve mention. In 2010, by detecting and utilizing vacant holes of a packing configuration, Huang and Ye (2010) proposed the greedy vacancy search algorithm for PECS, which improved the previous best-known results for 42 instances with $N \leq 200$. In 2012, M'Hallah and Alkandari (2012) devised a variable neighborhood search algorithm for PESC and reported experimental results for $N \leq 55$. In 2019, Bagattini et al. (2019) presented a clustering-based heuristic algorithm for PESC and reported three new best-known results for $N = 83$, 96, 109. Based on the method of Nurmela and Östergård (1997), Amore and Morales (2022) recently proposed a two-phase heuristic algorithm for PECS and reported results on seven selected large-scale instances in the range of $254 \leq N \leq 9996$.

Most notably, the popular Packomania website (Specht (2021)) shows that over the past 20 years many researchers continually improved the best-known results for PECS and PESC, even though many algorithms used to find these improved solutions were not published. For example, a special packing program called Packntile from the Pack'n'tile contest (`http://www.algit.eu/htmlji/Packntile/Packing_Contest_01052010.html`) provided the best-known results for a number of PECS and PESC instances. Kolossváry and Bowers (2010) improved the best-known PESC results by means of the hidden-force algorithm for $N = 83$, 94, 95. Cantrell provided the best-known solutions for many PECS and PESC instances (see Specht (2021)). The Packomania website (Specht (2021)), which provides a detailed and updated history of PECS and PESC problem results, discloses that the best-known results are continually being improved. This suggests that there is still room for further improvement, in spite of the fact that a large number of approaches have been proposed and tested. In this study, our goal is to advance the state-of-the-art by introducing an effective hybrid algorithm for PECS and PESC problems.

3. Modeling of the PECS and PESC problems

As noted, the constrained optimization PECS and PESC packing problems are difficult to tackle by means of popular local search approaches. Nevertheless, the objective functions of PECS and PESC can be regarded as strictly monotonic with respect to the container size L. Thus, by fixing the value of L, we can use the penalty function method to convert these constrained optimization problems into a series of unconstrained optimization subproblems with L values that gradually become smaller, and then solve these unconstrained subproblems via an unconstrained optimization method. With this in mind, the following subsections provide an unconstrained formulation for PECS and PESC in which the container size L is fixed.

3.1. PECS with a fixed container size L (PECS-L)

The unconstrained subproblems (denoted by PECS-L) focus on seeking a non-overlapping packing configuration of N circles in a square container with a fixed-size L. Following the popular penalty function approach used for the circle packing problems (Huang and Ye (2010, 2011), Lai et al. (2022b)) and the geometry optimization of clusters (Martínez et al. (2009)), we employ overlapping depths between the circles and between the circles and container borders to define the objective function of the unconstrained PECS subproblems. Given a fixed size L of a square container and a candidate packing configuration X defined by the coordinate vector $X = (x_1, y_1, \ldots, x_N, y_N)$ of N circles, the objective function $E_L(X)$ of the PECS-L subproblem with fixed L is defined as follows.

$$E_L(X) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} l_{ij}^2 + \sum_{i=1}^{N} (l_{ix}^2 + l_{iy}^2) \tag{10}$$

with

$$l_{ij} = max\{0, 2 - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}\} \tag{11}$$

$$l_{ix} = max\{0, |x_i| + 1 - \frac{L}{2}\} \tag{12}$$

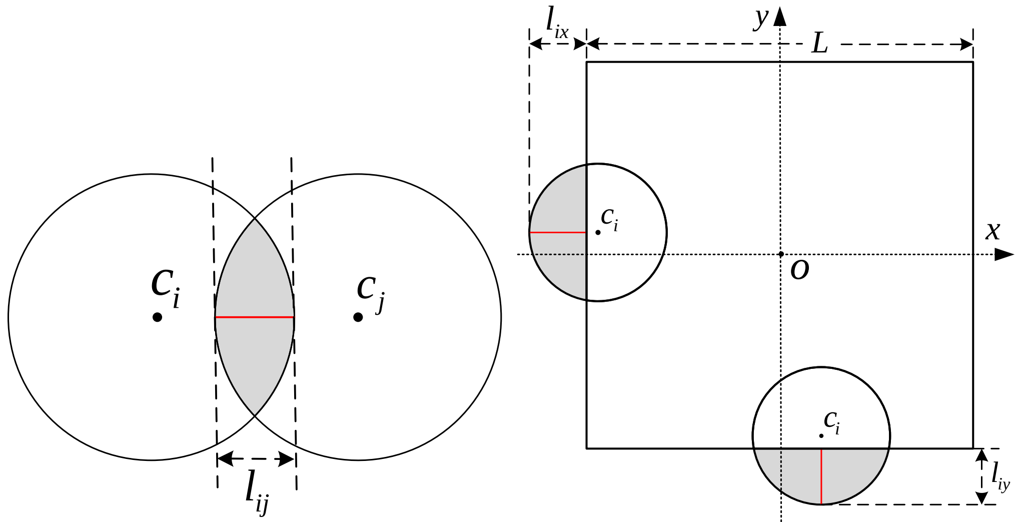
$$l_{iy} = max\{0, |y_i| + 1 - \frac{L}{2}\} \tag{13}$$

where l_{ij} represents the overlapping depth between two circles c_i and c_j, l_{ix} is the overlapping depth between a circle c_i and the vertical border of the container, and l_{iy} is the overlapping depth between a circle c_i and the horizontal border of the container. Fig. 1 illustrates these three categories of possible overlaps for an infeasible packing configuration.

The problem defined by Eqs. (10)-(13) is an unconstrained minimization problem whose objective function $E_L(X)$ measures the degree of non-overlapping constraint violation by the candidate solution X. As a result, X is a feasible solution if $E_L(X) = 0$, and else X is an infeasible solution.

3.2. PESC with a fixed container size L (PESC-L)

The PESC problem is a natural extension of the PECS problem in three-dimensional Euclidean space R^3. As for PECS, the corresponding subproblems with fixed L values (denoted by PESC-L) of PESC can be defined as follows.



(a) Overlapping depth l_{ij} between two circles c_i and c_j

(b) Two kinds of overlapping depths between a circle and the border of container

Figure 1 Possible overlaps in an infeasible packing configuration for PECS.

Let L be a given size of cube container. For a solution $X = (x_1, y_1, z_1, \ldots, x_N, y_N, z_N)$ with (x_i, y_i, z_i) being the center of the unit sphere s_i $(1 \leq i \leq N)$, the objective function $E_L(X)$ of PESC-L can be defined as:

$$E_L(X) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} l_{ij}^2 + \sum_{i=1}^{N} (l_{ix}^2 + l_{iy}^2 + l_{iz}^2) \tag{14}$$

where l_{ij} $(= max\{0, 2 - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}\})$ represents the overlapping depth between two unit spheres s_i and s_j, l_{ix} is the overlapping depth between a sphere s_i and the planes $x = \frac{L}{2}$ or $x = -\frac{L}{2}$, l_{iy} denotes the overlapping depth between a sphere s_i and the planes $y = \frac{L}{2}$ or $y = -\frac{L}{2}$, and l_{iz} represents the overlapping depth between a sphere s_i and the planes $z = \frac{L}{2}$ or $z = -\frac{L}{2}$. As for PECS-L, $E_L(X)$ measures the degree of constraint violation for the candidate solution X, and X is a feasible solution if $E_L(X) = 0$ and else X is an infeasible solution.

4. The Proposed PBTS Algorithm

Our proposed perturbation-based thresholding search algorithm for PECS and PESC is a stochastic optimization algorithm for solving the equal circle packing and equal sphere packing problems in a regular convex container. The main framework and the components of the algorithm are described in this section, employing the basic idea of handling the constrained PECS and PESC problems by solving a series of unconstrained optimization subproblems PECS-L and PESC-L with fixed decreasing L values.

4.1. Main framework of the proposed PBTS algorithm

Algorithm 1: Main framework of the PBTS algorithm

Input: Number of unit circles (or spheres) to be packed (N), maximum time limit (t_{max}), packing density of initial solution (p_0)

Output: The best packing found (X^*) and the size of container (L^b)

/* First Phase of the Search, where $E_L(X) < 10^{-25}$ implies that X is approximately feasible */
$p \leftarrow p_0$, $Flag \leftarrow 1$
$L \leftarrow \sqrt{\frac{N \times \pi}{p}}$ /* $L = \sqrt[3]{\frac{4 \times N \times \pi}{3 \times p}}$ for PESC */
$X \leftarrow RandomPacking(L)$
$X \leftarrow ThresholdSearch(X, L, Flag)$
while $E_L(X) < 10^{-25}$ **do**
  $p \leftarrow p + 10^{-3} \times rand(0,1)$
  $L \leftarrow \sqrt{\frac{N \times \pi}{p}}$ /* $L = \sqrt[3]{\frac{4 \times N \times \pi}{3 \times p}}$ for PESC */
  $X \leftarrow ThresholdSearch(X, L, Flag)$
end
$(X, L) \leftarrow AdjustContainer(X, L)$
$L^b \leftarrow L$, $X^* \leftarrow X$
/* Second Phase of the Search */
while *time()* $\leq t_{max}$ **do**
  $X \leftarrow RandomPacking(L^b)$
  $X \leftarrow ThresholdSearch(X, L^b, Flag)$ /* Algorithm 4 */
  if $E_L(X) < 10^{-25}$ **then**
    $(X, L) \leftarrow AdjustContainer(X, L^b)$ /* Algorithm 5 */
    if $L < L^b$ **then**
      $L^b \leftarrow L$, $X^* \leftarrow X$
    else
      $Flag \leftarrow (Flag + 1)\%2$
    end
  else
    $Flag \leftarrow (Flag + 1)\%2$ /* Change the type of perturbation */
  end
end

The PBTS algorithm is described in Algorithm 1, where X^* and L^b denote respectively the best feasible (i.e., non-overlapping) solution found so far and the corresponding container size (i.e., the side length of a square or cube container), and where X denotes the current solution. As shown in Algorithm 1, the procedure is composed of two search phases, in which the first phase aims to quickly find a high-density packing configuration by gradually reducing the container size and the second phase searches for the globally optimal solution by a multi-start approach. To ensure an effective search, the algorithm relies on a container adjustment method and a thresholding search procedure integrating a mixed perturbation strategy and local optimization.

In the first phase (lines 2–12), starting from an empirically estimated packing density p ($p = p_0$), the algorithm randomly generates a first initial packing configuration X in a container of size L, where L is determined according to the density p (lines 2–4). For the PECS problem, L is determined as $L = \sqrt{\frac{N\times\pi}{p}}$ since the packing density p of a feasible configuration with N unit circles in a square of size L is calculated as $p = \frac{N\times\pi\times r^2}{L^2}$, where r ($r = 1.0$) is the radius of unit circles. For the PESC problem, L is determined as $L = \sqrt[3]{\frac{4\times N\times\pi}{3\times p}}$, since the packing density p of a feasible configuration with N unit spheres in a cube of size L is calculated as $p = \frac{N\times\frac{4}{3}\times\pi\times r^3}{L^3}$. From this initial configuration X, a thresholding search procedure is applied to search for a non-overlapping packing configuration for the current L value by minimizing the objective function $E_L(X)$ defined in Section 3 (line 5). Subsequently, the search process enters a 'while' loop (lines 6–10) in which the packing density p is increased at each iteration by setting $p \leftarrow p + 10^{-3} \times rand(0,1)$ where $rand(0,1)$ denotes a random number between 0 and 1. Then L is recalculated according to the updated packing density p, and the current configuration X is further improved by the thresholding search procedure under the new L value. The 'while' loop terminates once the thresholding search method fails to find a non-overlapping solution for the current L value. The resulting configuration is then slightly adjusted by a container adjustment method to obtain a non-overlapping configuration for which the container size L is locally optimized, and the resulting solution and its L value are saved as X^* and L^b (lines 11–12).

In the second phase, the search executes a second 'while' loop until the time limit (t_{max}) is reached (lines 14–27). At each iteration, a random packing X is first generated in a container with the current best size L^b and then improved by the thresholding search procedure (lines 15–16). Subsequently, the container adjustment method is used to minimize the container size L once a feasible packing configuration X is obtained, while maintaining the configuration feasibility (line 18). The best configuration found so far X^* and the corresponding L^b are updated if the value of L^b is improved (line 20). To enhance the search robustness, the type of perturbation employed in the thresholding search procedure is then changed if the value of L^b has not been improved after the current thresholding search and container adjustment (lines 22 and 25). Finally, X^* and L^b are returned as the output of the algorithm.

4.2. Local optimization

Given an input solution X, to reach a nearest local minimum solution with respect to the objective function $E_L(X)$ in Section 3 and $U_\lambda(X, L)$ defined in Eq. (16), the thresholding search and container adjustment procedures both employ a popular and efficient quasi-Newton method called L-BFGS (Liu and Nocedal (1989)). In particular, the L-BFGS method employed by the proposed algorithm uses a line search method proposed by Hager and Zhang (2005) and is in most cases able to reach a high precision of 10^{-13} with respect to the maximum norm $||g||_\infty$ of gradient g of the objective function.

4.3. Perturbation operators

To ensure efficiency and generality, the thresholding search procedure (Section 4.4) employs two complementary and general-purpose perturbation operators to jump out of local optimum traps. The first is called the uniformly random perturbation (URP) which is widely used in the field of global optimization (Addis et al. (2008), Doye et al. (2004), Leary (2000), Wales and Doye (1997)). The second is called the sequential random perturbation (SRP) proposed in this study. These two perturbation operators are adaptively employed in the optimization process to enhance robustness (see Algorithm 1).

Algorithm 2: Uniformly random perturbation

Function $URP()$

Input: Input solution $X = (X[1], X[2], \ldots, X[n])$, number of variables n ($n = 2N$ for PECS and $3N$ for PESC), perturbation strength η_0

Output: The perturbed solution X

for $i \leftarrow 1$ ***to*** n **do**

    $X[i] \leftarrow X[i] + \eta_0 \times rand(-1, 1)$ `/* Perturb each component of` X `*/`

end

4.3.1. Uniformly random perturbation.

As shown in Algorithm 2, the URP operator drastically modifies the current solution in an uniformly random fashion. Given a candidate solution $X = (X[1], X[2], \ldots, X[n])$ with n variables, the URP operator shifts each component $X[i]$ of X in an interval $(-\eta_0, \eta_0)$, (i.e., $X[i] \leftarrow X[i] + rand(-\eta_0, \eta_0)$, $1 \le i \le n$), where $rand(-\eta_0, \eta_0)$ represents a uniformly distributed random number in the interval $(-\eta_0, \eta_0)$ and η_0 is a parameter called the perturbation strength and is set to 0.8 by default according to the study of Grosso et al. (2010).

4.3.2. Sequential random perturbation. Unlike the URP operator and most perturbation strategies in the literature, the SRP operator (described in Algorithm 3) consists of a series of small random perturbations, where each small perturbation is followed by a very short local optimization. Specifically, to obtain a high-quality offspring solution, the SRP operator performs consecutively I_{max} iterations (lines 3–15), each being composed of a small URP operation (lines 4–6) and a constant step gradient descent method consisting of only m steps (lines 7–12), where m is a parameter. The perturbation strength (η) is gradually decreased by a factor β as the number of iterations increases (line 13).

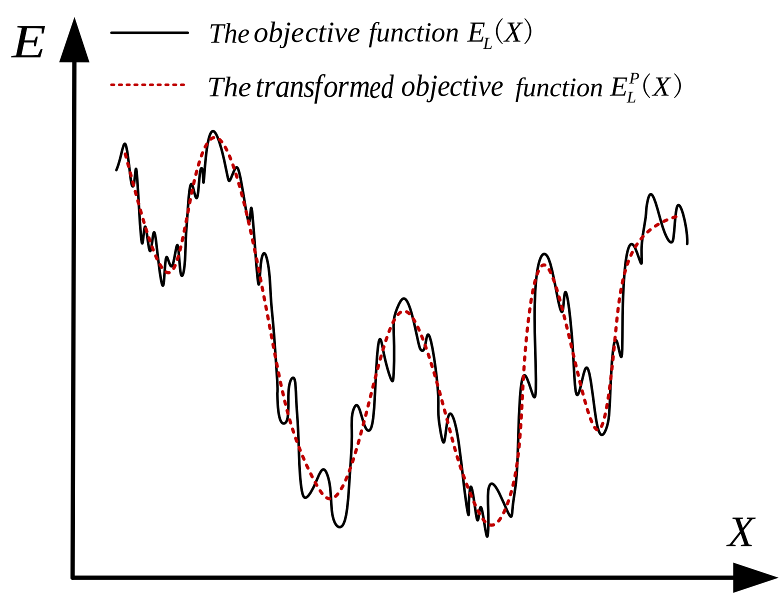


Figure 2 **A schematic diagram illustrating the effects of the sequential random perturbation in the one-dimensional case. The black line is the original potential energy surface $E_L(X)$ of the objective function and the red dotted line represents the surface ($E_L^P(X)$) transformed by the SRP perturbation operator. Note that the transformation of potential energy surface does not change the global optimum solution of the problem.**

Algorithm 3: Sequential random perturbation

Function $SRP()$

Input: Input solution $X = (X[1], X[2], \dots, X[n])$, number of variables n ($= 2N$ for PECS and $3N$ for PESC), perturbation strength η, parameters I_{max}, m, σ, $\beta \in (0,1)$

Output: The perturbed solution X

$I \leftarrow 0$

while $I < I_{max}$ **do**
    for $i \leftarrow 1$ ***to*** n **do**
        $X[i] \leftarrow X[i] + \eta \times rand(-1,1)$ `/* Perturb each coordinate of` X `*/`
    end
    for $m' \leftarrow 1$ ***to*** m **do**
        $g \leftarrow gradient(E, X)$ `/* Calculate the gradient of function` $E(X)$ `*/`
        for $i \leftarrow 1$ ***to*** n **do**
            $X[i] \leftarrow X[i] - \sigma \times \eta \times \frac{g[i]}{||g||}$
        end
    end
    $\eta \leftarrow \beta \times \eta$ `/* Reduce the perturbation strength by a factor` β `*/`
    $I \leftarrow I + 1$
end

The SRP operator is designed to skip low-quality local optimal solutions by using a number of small perturbations and the subsequent short local optimizations. As a result, the landscape of the objective function $E_L(X)$ is in effect smoothed by eliminating many small barriers. Figure 2 gives a schematic illustration of the expected effect of the SRP operator on the landscape of objective function $E_L(X)$. By this effect, the SRP operator helps the algorithm to sample a smaller and more promising set of local minima, and benefits the global optimization of function $E_L(X)$ in a number of cases, without changing the global optimum solution of the problem.

It is worth noting that the SRP operator does not require problem-specific knowledge and is a general-purpose method. Thus, it can be applied to the global optimization of any non-convex functions with a first-order derivative. The computational experiments in Section 6.1 show that this perturbation operator is very useful to enhance the effectiveness and efficiency of the algorithm for many instances and significantly outperforms some popular perturbation operators.

4.4. Thresholding search method

Algorithm 4: Thresholding search method for the unconstrained global optimization

Function *ThresholdSearch*
Input: Input solution X, size L of container, shrinkage factor μ ($\mu < 1.0$), acceptance rate ρ ($\rho < 1.0$), the depth of search ($MaxIter$), perturbation type $Flag$, and the parameters T, η_{min} and η_{max}.
Output: The best solution found (X^b)
$X \leftarrow Local\_Optimization(X)$ /* Minimize function $E_L(\cdot)$ from X */
$X^b \leftarrow X$
$NoImprove \leftarrow 0$, $t \leftarrow 0$, $Th_E \leftarrow 10^{-4}$, $N_{accept} \leftarrow 0$, $N_{reject} \leftarrow 0$ /* Initialization */
while $(NoImprove \leq MaxIter) \wedge (E_L(X^b) > 10^{-25})$ **do**
  $\eta \leftarrow \eta(t)$ /* Determine the strength of perturbation */
  if $Flag = 1$ **then**
    $X_{new} \leftarrow URP(X, \eta)$ /* Perform the URP perturbation */
  else
    $X_{new} \leftarrow SRP(X, \eta)$ /* Perform the SRP perturbation */
  end
  $X_{new} \leftarrow Local\_Optimization(X_{new})$ /* Minimize function $E_L(\cdot)$ */
  $\Delta_E \leftarrow E_L(X_{new}) - E_L(X)$
  if $(\Delta_E < Th_E) \wedge (\Delta_E \neq 0)$ **then**
    $X \leftarrow X_{new}$
    $N_{accept} \leftarrow N_{accept} + 1$
  else
    $N_{reject} \leftarrow N_{reject} + 1$
  end
  if $N_{accept} > \rho \times N_{reject}$ **then**
    $Th_E \leftarrow \mu * Th_E$ /* Decrease the threshold value Th_E */
  else
    $Th_E \leftarrow \frac{1}{\mu} * Th_E$ /* Increase the threshold value Th_E */
  end
  if $E_L(X) < E_L(X^b)$ **then**
    $X^b \leftarrow X$ /* Save the best solution found */
    $NoImprove \leftarrow 0$
  else
    $NoImprove \leftarrow NoImprove + 1$
  end
  $t \leftarrow t + 1$
end

To search for the global optimal solution of a nonlinear non-convex objective function $E_L(X)$ defined in Section 3, our algorithm employs a thresholding search procedure which follows the general threshold accepting method (Dueck and Scheuer (1990), Moscato and Fontanari (1990)) and is a variant of the dynamic thresholding search method proposed by Lai et al. (2022b). The thresholding search procedure aims to find a feasible packing configuration of N unit circles (or spheres) for a fixed size (L) of square or cube container. In our approach, it executes both a local search and two complementary perturbations.

The thresholding search procedure (denoted by $ThresholdSearch$) is described in Algorithm 4. Starting with an input solution X, *ThresholdSearch* first obtains a local optimal solution of $E_L(X)$ by performing the L-BFGS local optimization method (Liu and Nocedal (1989)) and saves the obtained solution as the best solution found so far (X^b) (lines 2–3). Subsequently, the thresholding search procedure performs a 'while' loop until the current best solution X^b cannot be improved for $MaxIter$ consecutive iterations or a feasible packing configuration X (i.e., $E_L(X) < 10^{-25}$) is found, where $MaxIter$ is a parameter called the search depth (lines 4–32).

At each iteration, the current solution X is first perturbed by the URP or SRP operator (see Section 4.3) according to the parameter $Flag$ with the perturbation strength η determined as follows. For the URP operator, the strength η is set to a constant η_0. For the SRP operator, to ensure a good tradeoff between search intensification and diversification, η is fixed by the following periodic function with respect to the number t of iterations:

$$\eta(t) = \eta_{min} + \frac{\eta_{max} - \eta_{min}}{2} \times [1 + \sin(t \times \frac{2\pi}{T})] \tag{15}$$

where T is the period that is set to 72 in this work, and η_{min} and η_{max} represent respectively the minimum and maximum values of $\eta(t)$. Fig. 3 shows the change of $\eta(t)$ as the number of iterations (t), which discloses how the value of $\eta(t)$ changes periodically in $[\eta_{min}, \eta_{max}]$ as a function of number of iterations. Note that the function $\eta(t)$ is a simple variant of the trigonometric function $sin(t)$ that is a well-known periodic function, where the parameters η_{min} and η_{max} were empirically determined.

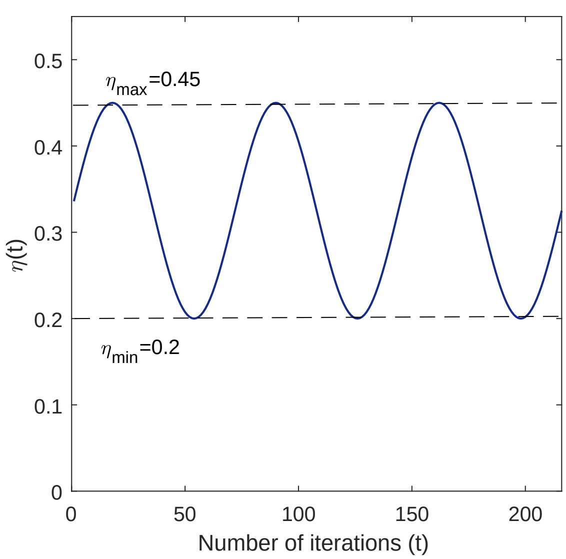


Figure 3 The change of strength of perturbation (η) as a function of the number of iterations.

Following this, the perturbed solution X is improved by the L-BFGS local optimization method. The newly obtained offspring solution X_{new} is accepted as the current solution if the gap between the objective values of X_{new} and X (i.e., $\Delta_E = E_L(X_{new}) - E_L(X)$) is smaller than a threshold value (Th_E) that is adaptively adjusted according to the current acceptance rate. Specifically, Th_E is decreased by multiplying by a factor $\mu \in (0,1)$ if $N_{accept} > \rho \times N_{reject}$, and is increased by dividing by μ otherwise, where N_{accept} represents the number of current acceptances, N_{reject} represents the number of current rejections, and ρ is a parameter representing the expected acceptance rate (hence a larger ρ value supports a higher acceptance rate for the offspring solutions).

Compared with the dynamic thresholding search method in Lai et al. (2022b), the current thresholding search procedure has two noteworthy features. First, it relies on two complementary perturbations that are applied adaptively and a periodic function is used to determine dynamically the perturbation strength for the SRP operator to reach a desirable tradeoff between search intensification and diversification. Second, the acceptance rate for new solutions determined by the parameter ρ, makes it possible to control the search behavior of the procedure.

Finally, it should be emphasized that this thresholding search method is of a general nature and can be applied to the global optimization of other non-convex differentiable functions, such as solving systems of nonlinear equations (Pei et al. (2019)) and optimizing general nonlinear functions (Bierlaire et al. (2010)).

4.5. Packing refinement by container adjustment

Algorithm 5: Container adjustment method

```
Function AdjustContainer
  Input: Input solution s_0 = (X_0, L_0), maximum number of iterations K (= 15)
  Output: The feasible local optimum configuration s = (X, L)
X ← X_0, L ← L_0, λ ← 10^6
for i ← 1 to K do
  | (X, L) ← Local_Optimization(U_λ, X, L)        /* Minimize U_λ(X, L) using L-BFGS */
  | λ ← 5 × λ
end
```

The container adjustment method is another main component of the proposed algorithm, whose goal is to slightly adjust the size L of the container and the coordinates of N circles (or spheres) for a given packing configuration such that the resulting solution becomes feasible while the size L of container is locally minimized.

Container adjustment can also be considered as a problem of finding a local minimum for a constrained optimization problem, starting from a given solution. In this study, we employ the sequential unconstrained minimization technique (SUMT) (Fiacco and McCormick (1964)). The SUMT method first converts the constrained optimization problem to a sequence of unconstrained problems, and then solves them in order until a feasible local minimum is reached. For the studied circles and sphere packing problems, the corresponding constrained optimization problems (Eqs. (1)-(4)) are converted into the following unconstrained optimization problems.

$$Minimize \quad U_\lambda(X,L) = \frac{L^2}{\lambda} + E(X,L) \tag{16}$$

where λ is a penalty factor and its each value λ_0 defines an unconstrained optimization function $U_{\lambda_0}(X,L)$, and L is a variable representing the size of a square or cube container. The term $E(X,L)$ containing $2N+1$ variables (or $3N+1$ variables for the PESC problem) measures the degree of constraint violations and can be written as follows:

$$E(X,L) = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} l_{ij}^2 + \sum_{i=1}^{N}(l_{ix}^2 + l_{iy}^2 + l_{iz}^2) \tag{17}$$

where l_{ij} represents the overlapping depth between two unit circles (or spheres) c_i and c_j, and l_{ix}, l_{iy}, and l_{iz} represent respectively the overlapping depths between a unit circle (or sphere) c_i and the border of the container in the different coordinate axes (see Section 3 for the detailed description). For the PECS problem in two-dimensional Euclidean space R^2, l_{iz} should be omitted.

The container adjustment method is described in Algorithm 5. Starting from an input solution (X_0, L_0) and an initial λ value (set to 10^6), the procedure performs K iterations to obtain a feasible packing configuration in which the size L of container is locally minimized. At each iteration, the L-BFGS method is applied to minimize $U_\lambda(X,L)$, and then the value of λ is increased by a factor of 5 and the resulting solution is used as the input solution of the next iteration. Eventually, a feasible packing configuration (X,L) with $E(X,L)=0$ will be obtained when the penalty factor λ reaches a very large value.

The container adjustment method described here is a simple variant of previous ones used in (Huang and Ye (2010), Lai et al. (2022b)), which have been shown to be very efficient for circle packing problems. It is worth noting that the present container adjustment method is able to reach a high-precision result and is much more robust than the popular bisection method in (Huang and Ye (2011)) since the occasional failure of the local optimization procedure (i.e., L-BFGS) in the bisection method will result in an inaccuracy of the result.

5. Computational Experiments and Assessments

In this section, we present extensive computational experiments to assess the performance of the proposed algorithm for solving the PECS and PECS problems. The assessment is based on well-known benchmark instances and comparisons with the best-known results shown at the Packomania website (Specht (2021)).

The codes of the proposed PBTS algorithm, the best solutions found in this work, and the detailed computational results of algorithm are available from the online supplement of the paper (Lai et al. (2022a)).

5.1. Parameter Settings and Experimental Protocol

Table 3 Settings of parameters

Parameters	Section	Description	Values
p_0	4.1	packing density of initial solution	$\{0.5, 0.7\}$
$MaxIter$	4.4	search depth of thresholding search	400
μ	4.4	parameter used to tune the threshold Th_E	0.6
ρ	4.4	acceptance rate of solutions for thresholding search	0.5
I_{max}	4.3	number of iterations for SRP perturbation	50
m	4.3	parameter in SRP perturbation operator	5
σ	4.3	coefficient used in SRP perturbation	4.0
β	4.3	coefficient used in SRP perturbation	0.99
η_0	4.3.1	strength of URP perturbation	0.8
η_{max}	4.4	maximum strength of SRP perturbation	$\{0.45, 0.8\}$
η_{min}	4.4	minimum strength of SRP perturbation	$\{0.1, 0.2\}$

The parameters employed by our algorithm are indicated in Table 3. The default settings of these parameters were empirically determined via preliminary experiments. The values of parameter p_0 representing the packing density of initial solutions were empirically set to 0.5 and 0.7 for PESC and PECS, respectively. The values of parameters η_{min} and η_{max} were respectively set to 0.2 and 0.45 for PECS, and 0.1 and 0.8 for PESC. The default settings for the other parameters are given in Table 3.

Our algorithm was implemented in the C++ language and was executed on a computer with an Intel(R) Xeon (R) Platinum 9242 CPU (2.3 GHz), and running under a Linux operating system. Due to its stochastic nature, the algorithm was independently performed multiple times with different random seeds for each tested instance to assess the overall and average performance of the algorithm. For the PECS problem, the algorithm was independently executed 20 times for each instance in the range of $2 \leq N \leq 400$. The maximum time limit (t_{max}) per run was set according to the instance size (N): 2 hours for $N \leq 100$, 4 hours for $101 \leq N \leq 200$, 8 hours for $201 \leq N \leq 300$, and 12 hours for $301 \leq N \leq 400$. These time limits are consistent with those used in (Huang and Ye (2010)) to ensure a meaningful comparison between the proposed algorithm and the state-of-the-art algorithms in the literature. For the PESC problem, our algorithm was independently executed 10 times for each instance in the range of $2 \leq N \leq 200$, and the maximum time limit per run t_{max} was set to 1 hour for $N \leq 50$, 4 hours for $51 \leq N \leq 100$, and 10 hours for $101 \leq N \leq 200$, given that the PESC problem contains more decision variables than the PECS problem for a same N value.

5.2. Computational results and comparison on the PECS instances

Tables 4–6 summarize the computational results of the algorithm on the PECS problem in the range of $2 \leq N \leq 400$. Table 4 reports the detailed results for 40 representative PECS instances, while the detailed computational results for the remaining instances are available in the online supplement of the paper (Lai et al. (2022a)). The first and second columns of Table 4 indicate the size of instances (N) and the best-known results (L^*) in terms of the objective value, and the results of our algorithm are shown in columns 3–8, including the best objective value (L_{best}) over 20 independent runs, the average objective value (L_{avg}), the worst objective value (L_{worst}), the difference between L_{best} and L^*, the success rate (SR) of obtaining the best objective value (L_{best}), and the average running time in seconds for each run of the algorithm to obtain its final result ($time(s)$). For L_{best}, L_{avg}, L_{worst}, the improved results are indicated in bold compared to the best-known result L^*, and the worse results are indicated in italics. In addition, the last three rows of the table indicate the numbers of instances for which the proposed algorithm obtained a better, equal, or worse result compared to the best-known result L^* in terms of L_{best}, L_{avg}, and L_{worst}.

Table 4 shows that the proposed algorithm is very efficient compared to the state-of-the-art results in the literature. In terms of L_{best}, the algorithm improves and matches respectively the best-known results for 16 and 23 out of the 40 instances, and misses the best-known result only for one instance. The average result L_{avg} of the algorithm is better than and equal to the best-known result L^* respectively for 16 and 19 instances. It is worth noting that even the worst result L_{worst} over 20 runs of the algorithm improves and matches

Table 4 Computational results and comparison on 40 representative PECS instances in the range of $2 \leq N \leq 400$. In terms of L_{best}, L_{avg} and L_{worst}, results better than the best-known results L^* are indicated in bold and worse results are indicated in italics.

		PBTS (this work)					
N	L^* (Specht (2021))	L_{best}	L_{avg}	L_{worst}	$L_{best} - L^*$	SR	$time(s)$
91	18.692734847	18.692734847	18.692734847	18.692734847	0.0	20/20	220
92	18.755713984	18.755713984	18.755713984	18.755713984	0.0	20/20	1024
93	18.894150540	18.894150540	18.894150540	18.894150540	0.0	20/20	71
94	18.941057478	18.941057478	18.941057478	18.941057478	0.0	20/20	243
95	19.076554913	19.076554913	19.076554913	19.076554913	0.0	20/20	359
96	19.129447365	19.129447365	19.129447365	19.129447365	0.0	20/20	150
97	19.188408788	19.188408788	19.188408788	19.188408788	0.0	20/20	233
98	19.218577371	19.218577371	19.218577371	19.218577371	0.0	20/20	348
99	19.238684303	19.238684303	19.238684303	19.238684303	0.0	20/20	37
100	19.454847253	19.454847253	19.454847253	19.454847253	0.0	20/20	326
191	26.635323947	26.635323947	26.635323947	26.635323947	0.0	20/20	490
192	26.706309251	26.706309251	26.706309251	*26.706309254*	0.0	19/20	4035
193	26.792374948	*26.792374956*	*26.792393690*	*26.792540178*	8.23E-09	2/20	10165
194	26.840126463	**26.836480274**	**26.837233280**	**26.838362788**	-3.65E-03	12/20	3250
195	26.872063371	26.872063371	26.872063371	26.872063371	0.0	20/20	574
196	26.992467225	26.992467225	26.992467225	26.992467225	0.0	20/20	414
197	27.121091211	27.121091211	27.121091211	27.121091211	0.0	20/20	4324
198	27.194835943	27.194835943	*27.194845501*	*27.194857182*	0.0	11/20	4335
199	27.272163202	27.272163202	*27.272163226*	*27.272163412*	0.0	11/20	6157
200	27.312853154	27.312853154	*27.323213491*	*27.327787437*	0.0	2/20	6734
291	32.737055178	**32.713223771**	**32.713223771**	**32.713223771**	-2.38E-02	20/20	10660
292	32.784245264	**32.771377288**	**32.771377288**	**32.771377288**	-1.29E-02	20/20	3335
293	32.803781063	**32.803747327**	**32.803747329**	**32.803747333**	-3.37E-05	9/20	5671
294	32.827175126	32.827175126	32.827175126	32.827175126	0.0	20/20	5856
295	32.847220793	32.847220793	32.847220793	32.847220793	0.0	20/20	13273
296	32.939976201	**32.939965541**	**32.939965541**	**32.939965541**	-1.07E-05	20/20	9668
297	32.991738983	**32.990170865**	**32.990170865**	**32.990170865**	-1.57E-03	20/20	5128
298	33.024730665	33.024730665	33.024730665	33.024730665	0.0	20/20	4681
299	33.058743039	**33.058648838**	**33.058648838**	**33.058648838**	-9.42E-05	20/20	6103
300	33.091154939	33.091154939	33.091154939	33.091154939	0.0	20/20	7146
391	37.786012401	**37.769753050**	**37.769763033**	**37.769777333**	-1.63E-02	5/20	29084
392	37.861151301	**37.836566990**	**37.837784516**	**37.839320024**	-2.46E-02	1/20	34452
393	37.919186895	**37.899488243**	**37.899704475**	**37.900021701**	-1.97E-02	3/20	31459
394	37.930436797	**37.930272137**	**37.930278074**	**37.930310285**	-1.65E-04	4/20	24346
395	37.962314535	**37.960263101**	**37.960273664**	**37.960474373**	-2.05E-03	19/20	14013
396	37.975202348	37.975202348	*37.975627191*	*37.983699109*	0.0	18/20	25958
397	38.023361885	**38.023245990**	**38.023250121**	**38.023252875**	-1.16E-04	8/20	23505
398	38.082889973	**38.053496853**	**38.066853125**	**38.082490840**	-2.94E-02	4/20	11965
399	38.148780581	**38.128423455**	**38.128423455**	**38.128423455**	-2.04E-02	20/20	7370
400	38.164523000	**38.164286993**	**38.164286993**	**38.164286993**	-2.36E-04	20/20	9305
#Improved		16	16	16			
#Equal		23	19	18			
#Worst		1	5	6			

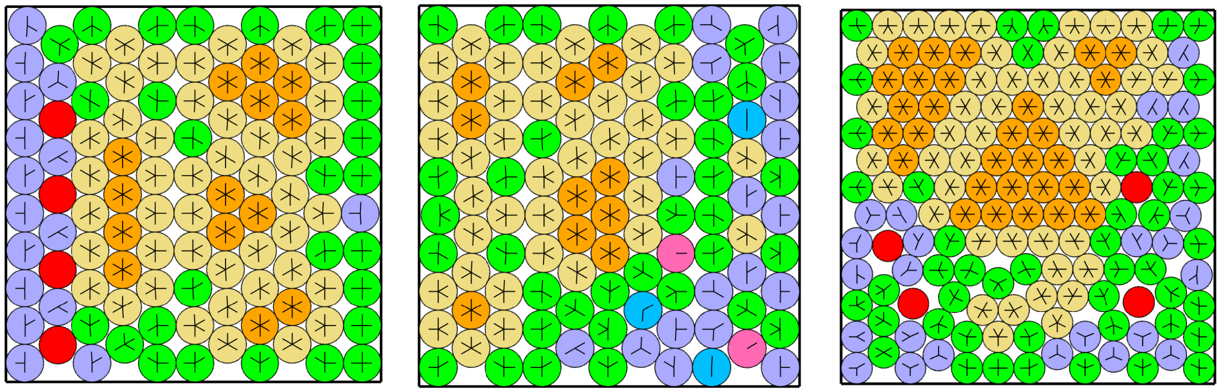
Table 5 Summary of computational results for the PECS problem

N	#Instance	#Improve	#Equal	#Worse
2-50	49	0	49	0
51-100	50	0	50	0
100-150	50	1	49	0
151-200	50	7	42	1
201-250	50	25	25	0
251-300	50	35	15	0
301-350	50	45	5	0
351-400	50	43	7	0
Total	399	156	242	1

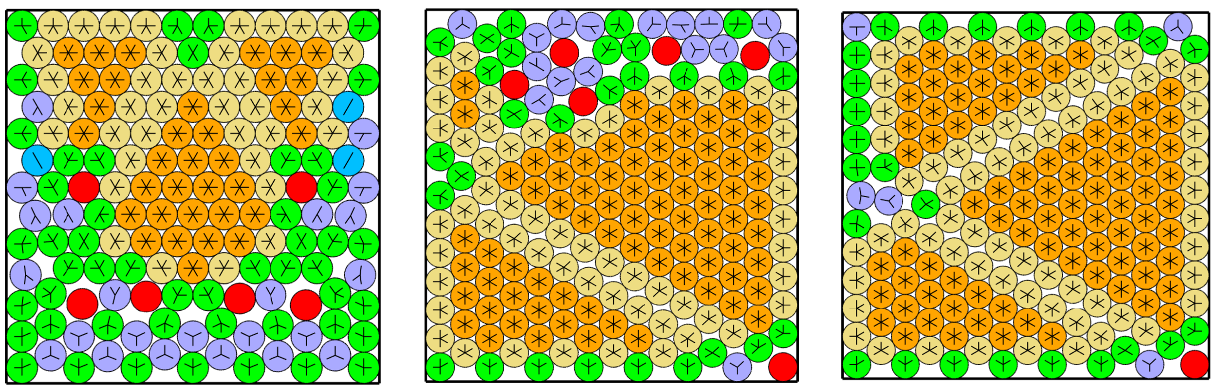
Table 6 Improved results for 156 instances of PECS in the range of $N \leq 400$ in terms of L_{best}, compared to the best-known results (BKR) in the literature.

N	BKR	L_{best}	N	BKR	L_{best}	N	BKR	L_{best}
106	19.993921681	**19.993921677**	275	31.760669189	**31.727696959**	342	35.353034567	**35.349992599**
153	24.010218767	**24.004609758**	277	31.876485279	**31.875605745**	343	35.444701701	**35.424528912**
157	24.259862394	**24.259862387**	278	31.904139803	**31.896621548**	344	35.501634171	**35.494954267**
170	25.252698973	**25.252697180**	282	32.271825992	**32.247367828**	345	35.565061674	**35.557861904**
171	25.325602865	**25.325594655**	283	32.349428155	**32.328946505**	346	35.620800461	**35.615716868**
172	25.417214968	**25.404460665**	284	32.371743196	**32.371671534**	347	35.658729029	**35.653297131**
181	25.985618408	**25.975260489**	285	32.442795717	**32.440898585**	348	35.685807919	**35.685495033**
194	26.840126463	**26.836480274**	286	32.453648756	**32.453626290**	349	35.719505469	**35.718924303**
205	27.576876117	**27.567250117**	288	32.572406278	**32.571998609**	350	35.771493660	**35.771461077**
207	27.677311798	**27.677311693**	290	32.658281934	**32.619667676**	351	35.793678470	**35.793675756**
210	27.865431130	**27.865264940**	291	32.737055178	**32.713223771**	352	35.952656162	**35.915743200**
215	27.999469469	**27.999457017**	292	32.784245264	**32.771377288**	353	35.995094545	**35.984799658**
216	27.999894488	**27.999890345**	293	32.803781063	**32.803747326**	354	36.006531662	**36.006336175**
218	28.396275192	**28.396274204**	296	32.939976201	**32.939965541**	355	36.098028799	**36.075573954**
219	28.505470621	**28.503438714**	297	32.991738983	**32.990170865**	356	36.125768096	**36.098038505**
220	28.581964018	**28.579623308**	299	33.058743039	**33.058648838**	357	36.144514642	**36.143900174**
221	28.650217350	**28.647970386**	301	33.107664708	**33.107188751**	359	36.245440882	**36.214080980**
223	28.746373096	**28.744925481**	302	33.115483620	**33.114674076**	360	36.313349883	**36.296057544**
224	28.791478737	**28.791012002**	305	33.369469591	**33.365641905**	361	36.329623084	**36.316021050**
225	28.892529959	**28.872701621**	306	33.472130383	**33.433775371**	362	36.351806981	**36.351199205**
226	28.980195790	**28.977237925**	307	33.533342804	**33.528160031**	364	36.393238790	**36.367921257**
227	29.047402657	**29.043187997**	308	33.592291592	**33.581624679**	365	36.440462302	**36.430903871**
229	29.166477858	**29.142413853**	309	33.662242538	**33.653059584**	366	36.449293679	**36.448985622**
230	29.186674755	**29.186139367**	310	33.708969061	**33.706874743**	367	36.461059367	**36.461059364**
232	29.331629096	**29.331615175**	311	33.766968464	**33.763821954**	370	36.617447623	**36.616904172**
233	29.372019605	**29.371417294**	312	33.810134916	**33.800477552**	371	36.630690132	**36.630690061**
235	29.412549407	**29.412498235**	313	33.819660112	**33.819647499**	374	36.860238563	**36.860209804**
237	29.498377024	**29.498316339**	314	33.863166811	**33.863128095**	375	36.905797738	**36.905763282**
240	29.666279530	**29.666279350**	315	33.977361251	**33.977355640**	376	36.934713162	**36.933889065**
244	29.914182807	**29.912487768**	316	34.106569996	**34.066488590**	377	36.956938729	**36.951443202**
246	29.968323001	**29.967083687**	317	34.157653052	**34.139244391**	378	36.964726408	**36.964387688**
249	30.335969818	**30.330357801**	318	34.230632899	**34.212087493**	379	37.168995378	**37.163358097**
250	30.417031057	**30.413810109**	319	34.278012052	**34.244701317**	380	37.268203266	**37.254728252**
251	30.514477048	**30.488615243**	320	34.301272474	**34.296258944**	381	37.328685368	**37.317946628**
252	30.556359106	**30.554594854**	321	34.330158112	**34.304146089**	382	37.399833611	**37.367815912**
254	30.637242017	**30.629122311**	322	34.384694645	**34.363743605**	383	37.453237457	**37.427104603**
255	30.688409827	**30.688395200**	323	34.453936080	**34.443942400**	384	37.509762344	**37.480145181**
256	30.724745895	**30.724745891**	324	34.469293461	**34.469293437**	385	37.524483179	**37.520288824**
257	30.824174685	**30.818433601**	325	34.497557560	**34.495918525**	386	37.555106698	**37.553644453**
259	30.971015228	**30.959405025**	326	34.538708641	**34.530949661**	387	37.596693465	**37.561988045**
260	31.002117303	**30.976264045**	327	34.591603459	**34.591579616**	388	37.634842769	**37.626421391**
261	31.021231115	**31.020513868**	328	34.627862893	**34.627258700**	389	37.669261935	**37.668211363**
263	31.135387935	**31.135221398**	330	34.665648167	**34.665385656**	390	37.697840561	**37.696589756**
264	31.200792357	**31.198320781**	331	34.757395176	**34.757317756**	391	37.786012401	**37.769753050**
265	31.217512841	**31.217512377**	332	34.790927642	**34.790437779**	392	37.861151301	**37.836566990**
266	31.255856470	**31.252989666**	333	34.824467247	**34.824467058**	393	37.919186895	**37.899488243**
267	31.262745851	**31.262008812**	334	34.848927176	**34.848252392**	394	37.930436797	**37.930272137**
268	31.304614023	**31.290274210**	335	34.888861313	**34.888813154**	395	37.962314535	**37.960263101**
269	31.314762420	**31.314762419**	336	34.923285419	**34.923052382**	397	38.023361885	**38.023245990**
271	31.437717984	**31.437713367**	338	34.974622430	**34.974609760**	398	38.082889973	**38.053496853**
272	31.553295627	**31.553030915**	339	34.987295963	**34.987292522**	399	38.148780581	**38.128423455**
273	31.605207996	**31.605028606**	341	35.292676076	**35.276436460**	400	38.164523000	**38.164286993**

the best-known result for 16 and 18 instances, respectively. Moreover, the success rate is 100% for 25 out of the 40 instances, occurring notably for the cases with $N \leq 100$. It can be concluded that the proposed algorithm is very efficient compared to the previous PECS algorithms. By the way, we point out that according to our experiments the performance of the proposed algorithm does not depend on the formulation of the PECS problem, and



(a) Previous best packing for $N = 106$ (b) Improved packing for $N = 106$ (c) Previous best packing for $N = 153$

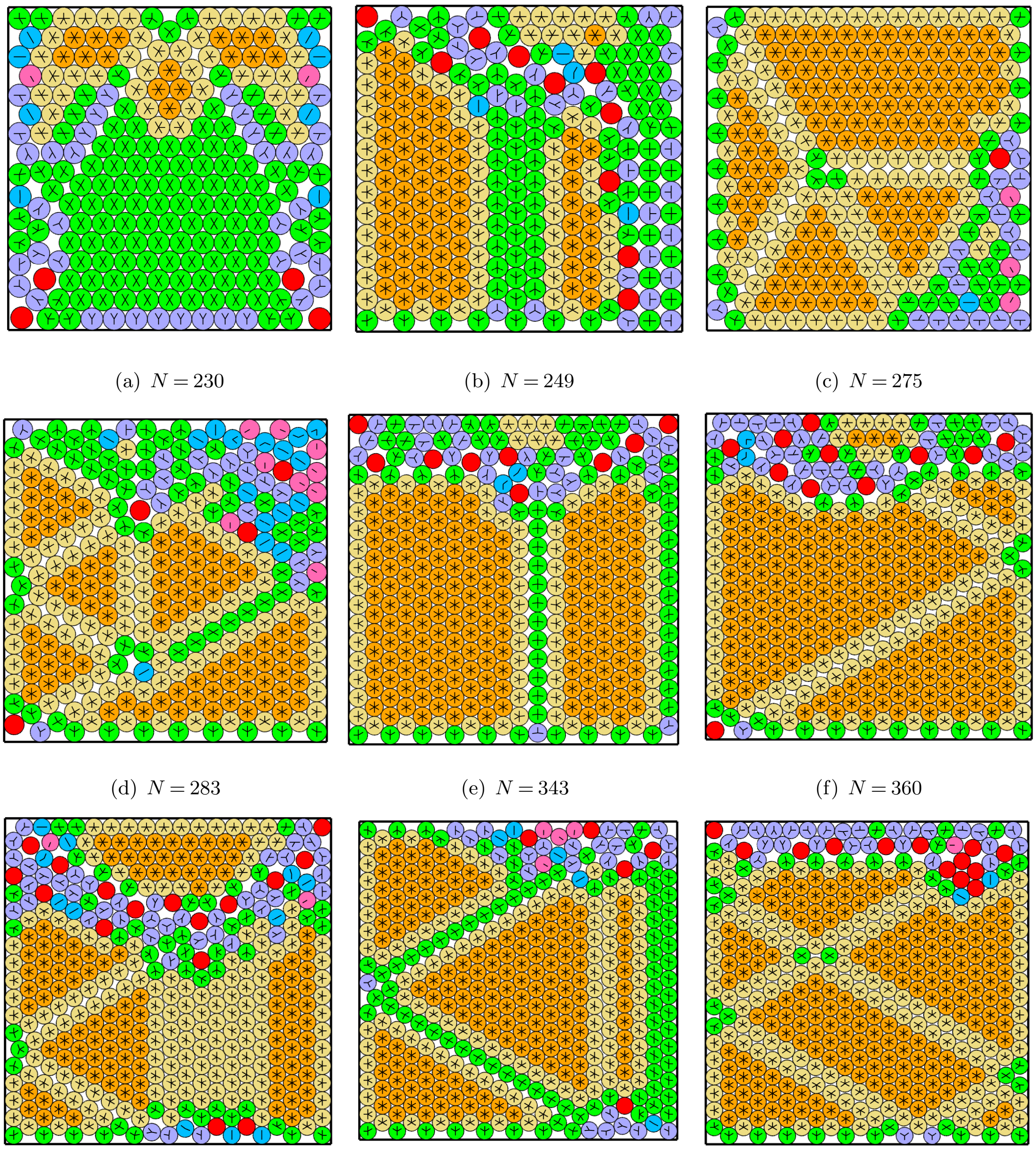


(d) Improved packing for $N = 153$ (e) Previous best packing for $N = 181$ (f) Improved packing for $N = 181$

Figure 4 **Comparisons between the previous best-known solution and the improved solution for 3 representative instances in the range of $N \leq 200$.**

our algorithm is able to obtain similar results with the equivalent formulation by Addis et al. (2008) mentioned in Section 1.

Tables 5 and 6 summarize the computational results for all 399 instances in the range of $2 \leq N \leq 400$. Table 5 shows that the proposed algorithm improves, matches and misses the best-known results respectively for 156, 242 and 1 instance. The improved results are summarized in Table 6, together the best-known results in the literature, which shows that for the widely studied instances in the literature (i.e., $N \leq 200$), our algorithm improves the best-known results for 8 instances and the smallest size is $N = 106$. For the 200 instances in the range of $201 \leq N \leq 400$, the algorithm improves the best-known results for 148 instances. These outcomes indicate that the proposed algorithm significantly outperforms the existing PECS algorithms in the literature.



(a) $N = 230$ (b) $N = 249$ (c) $N = 275$

(d) $N = 283$ (e) $N = 343$ (f) $N = 360$

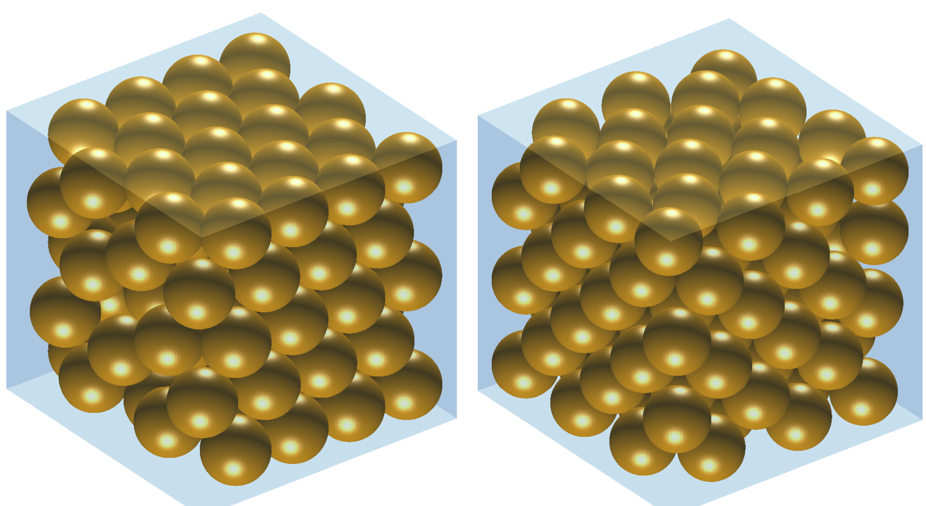
(g) $N = 382$ (h) $N = 388$ (i) $N = 400$

Figure 5 **Improved packing configurations found in this work for 9 representative PECS instances in the range of** $201 \leq N \leq 400$**.**

To provide a visual comparison between the previous best-known solutions and the improved solutions for some representative instances, Fig. 4 gives a graphical representation of the previous best-known solutions and the improved solutions of the three instances for $N = 106$, 153 and 181. For the instance with $N = 106$, the previous best-known result and the improved result respectively yield $L^* = 19.993921681$ and $L_{best} = 19.993921677$, and the difference between L_{best} and L^* is very small (i.e., $L_{best} - L^* = -3.74 \times 10^{-9}$). However, as observed from the subfigures (a) and (b) of Fig. 4, the difference between these two solutions in the graphical representation is significant, which means that a very small difference in the result can lead to two significantly different configurations and that there exist a number of configurations with very similar L values. For $N = 153$ and 181, the improved solution differs from the previous best-known solution by being more compact and symmetrical.

To give an intuitive impression about the nature of the best configurations found in this work, Fig. 5 provides the graphical representations for several representative best configurations. These representations disclose that the best configurations found in this work exhibit a variety of packing patterns.

5.3. Computational results and comparison on the PESC instances



(a) $N = 96$ (b) $N = 109$

Figure 6 Improved packing configurations for 2 representative PESC instances.

Table 7 Summary of computational results for the PESC problem

N	#Instance	#Improve	#Equal	#Worse
2-50	49	0	48	1
51-100	50	2	42	6
100-150	50	33	16	1
151-200	50	31	18	1
Total	199	66	124	9

Table 8 Improved best results for 66 instances of PESC in the range of $N \leq 200$, compared to the best-known results (BKR) in the literature (Specht (2021)).

N	BKR	L_{best}	N	BKR	L_{best}	N	BKR	L_{best}
94	8.923821830	**8.923821803**	132	9.900529520	**9.900168047**	174	10.731753582	**10.723893469**
96	8.956098398	**8.956012987**	134	9.969788469	**9.968398864**	175	10.747079787	**10.746999788**
101	9.071067812	**9.071067798**	135	9.997248155	**9.992767099**	176	10.769091939	**10.769085289**
102	9.071067812	**9.071067752**	136	10.005773959	**10.002501384**	177	10.797319352	**10.795952217**
109	9.256067536	**9.256055096**	137	10.009904202	**10.009483099**	178	10.874840404	**10.843408746**
110	9.315177255	**9.315142168**	138	10.013103535	**10.013103499**	179	10.906678902	**10.878347102**
111	9.350891923	**9.350777107**	139	10.013875394	**10.013871054**	180	10.908568331	**10.907954805**
112	9.381202299	**9.381195998**	145	10.182224825	**10.182224821**	181	10.934189362	**10.925071993**
113	9.432624730	**9.432624729**	146	10.234614828	**10.234607999**	182	10.945963366	**10.925072004**
114	9.466230073	**9.466077280**	147	10.261562986	**10.261561623**	183	10.948204573	**10.943111098**
115	9.511189528	**9.506114111**	148	10.300703719	**10.297021071**	184	10.949401271	**10.949347370**
116	9.521926074	**9.521920535**	149	10.326031998	**10.323831269**	185	10.954519536	**10.953766994**
117	9.536291586	**9.535542146**	150	10.341280996	**10.333464149**	186	10.955528631	**10.954992621**
118	9.539688701	**9.539355204**	151	10.365253238	**10.358713606**	187	10.956001897	**10.955754270**
119	9.541327956	**9.541325287**	152	10.374597071	**10.372956934**	191	11.064604714	**11.050914327**
120	9.542056732	**9.542033251**	154	10.397742319	**10.392876250**	192	11.071024807	**11.068578718**
123	9.647335796	**9.647281695**	155	10.432530776	**10.420260141**	193	11.071067812	**11.070025179**
124	9.655733621	**9.655730130**	157	10.450379879	**10.450379451**	194	11.071067812	**11.070959098**
125	9.656804408	**9.656804109**	158	10.466444211	**10.450379879**	197	11.183482546	**11.148837915**
127	9.734886581	**9.734698832**	159	10.468298279	**10.468298215**	198	11.216034385	**11.210995781**
129	9.831897328	**9.831895609**	160	10.475182230	**10.475182106**	199	11.249615229	**11.225541804**
131	9.877737172	**9.877735748**	161	10.479983040	**10.479983036**	200	11.271851411	**11.249879531**

To further check the effectiveness of our PBTS algorithm for the PESC problem, we ran the algorithm 10 times to solve each instance in the range of $2 \leq N \leq 200$. The summary results of the experiment are reported in Table 7 and the improved results are listed in Table 8. Detailed computational results are provided in the online supplement of the paper (Lai et al. (2022a)).

Table 7 indicates that the proposed PBTS algorithm is very competitive compared to the existing PESC algorithms, improving the best-known result for 66 out of the 199 instances, matching the best-known results on 124 instances, and missing the best-known result on only 9 instances. Interestingly, the proposed algorithm improves or matches the best-known results with greater frequency as the number (N) of spheres increases. Meanwhile, for our algorithm, the PESC problem is evidently more difficult than the PECS problem, as our algorithm fails to find the best-known results for 9 small instances of PESC. Thus, it is very likely that the best results found by our algorithm are suboptimal for many instances and there is still room for improvement.

Table 8 shows that the proposed algorithm is able to improve the best-known results for a large number of PESC instances but the improvement in the objective value is small. Nevertheless, the newly found solutions have configurations that differ appreciably from the previous best configurations. As shown in Section 5.2, a very small improvement (such as 10^{-8}) in the objective value can lead to a completely different packing configuration.

To give a further intuitive impression about the nature of the best solutions found in this work, Fig. 6 provides a graphical representation for the improved solutions of two representative PESC instances. According to the Packomania website (Specht (2021)), the best-known results of these two instances have been improved at different times by various methods, and the current best-known solutions were found by a recent clustering-based heuristic algorithm of Bagattini et al. (2019).

In summary, the fact that the proposed PBTS algorithm further improves the best-known results with a high success rate shows that the algorithm is remarkably effective for the PESC problem.

6. Analysis

We now turn to an analysis of two important elements of the PBTS algorithm, i.e., the perturbation operators and the two-phase search strategy. The sensitivity analysis of several important parameters is also provided in the online supplement (Lai et al. (2022a)).

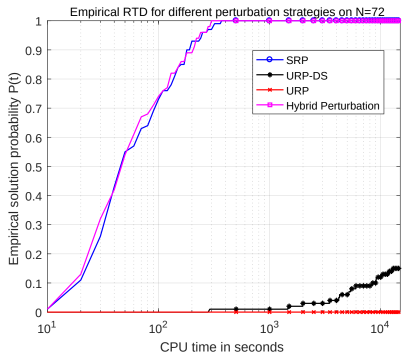
6.1. Importance of hybrid perturbation

To assess the importance of the hybrid perturbation by which thresholding search adaptively integrates the two different perturbation operators (SRP and URP), we compare the proposed algorithm with three algorithmic variants, two of which respectively employ only SRP or URP as its perturbation operator, and the third employs a variant of URP denoted URP-DS in which the perturbation strength dynamically change in an interval $[0.6, 1.0]$ as a function of the number of iterations ($\eta(t)$ in Eq. (15)). The proposed algorithm and its three variants were each executed 100 times to solve six representative instances.

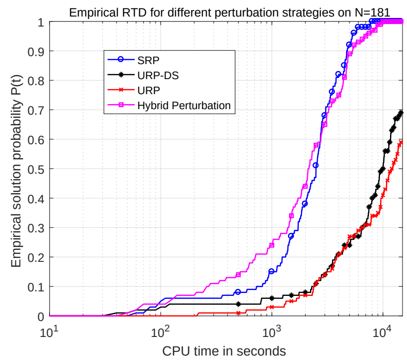
To analyze and compare the behaviors of these algorithms, we used the empirical run-time distribution (RTD) for stochastic optimization methods (Hoos and Stützle (2000)). For a given instance, the cumulative empirical RTD of an algorithm is a function P mapping the run-time t to the probability of obtaining the current best-known solution within time t. Specifically, the function P is defined as follows:

$$P(t) = \frac{|\{j : rt(j) \leq t\}|}{Q} \tag{18}$$

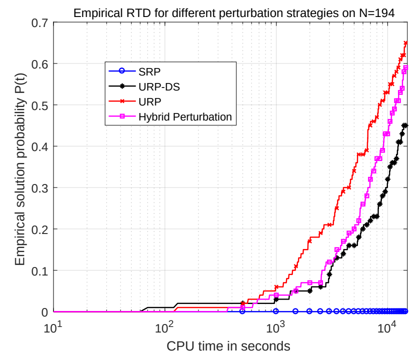
where $rt(j)$ is the running time of the j-th successful run to obtain the current best-known solution (i.e., the best packing configuration found in this work) and Q is the number of runs performed (where $Q = 100$ in this experiment). As demonstrated in (Hoos and Stützle



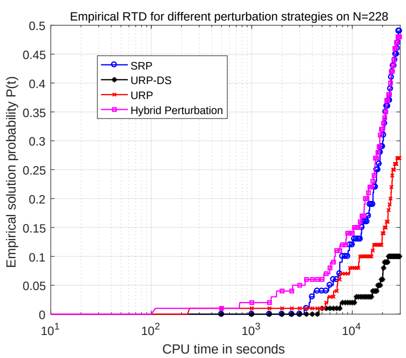
(a) $N = 72$ of PESC



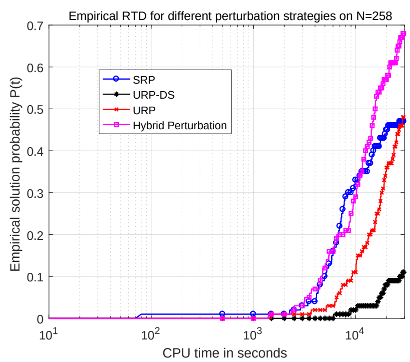
(b) $N = 181$ of PECS



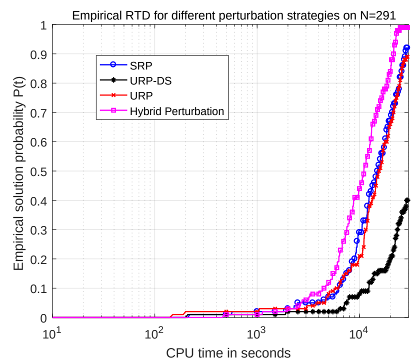
(c) $N = 194$ of PECS



(d) $N = 228$ of PECS



(e) $N = 258$ of PECS



(f) $N = 291$ of PECS

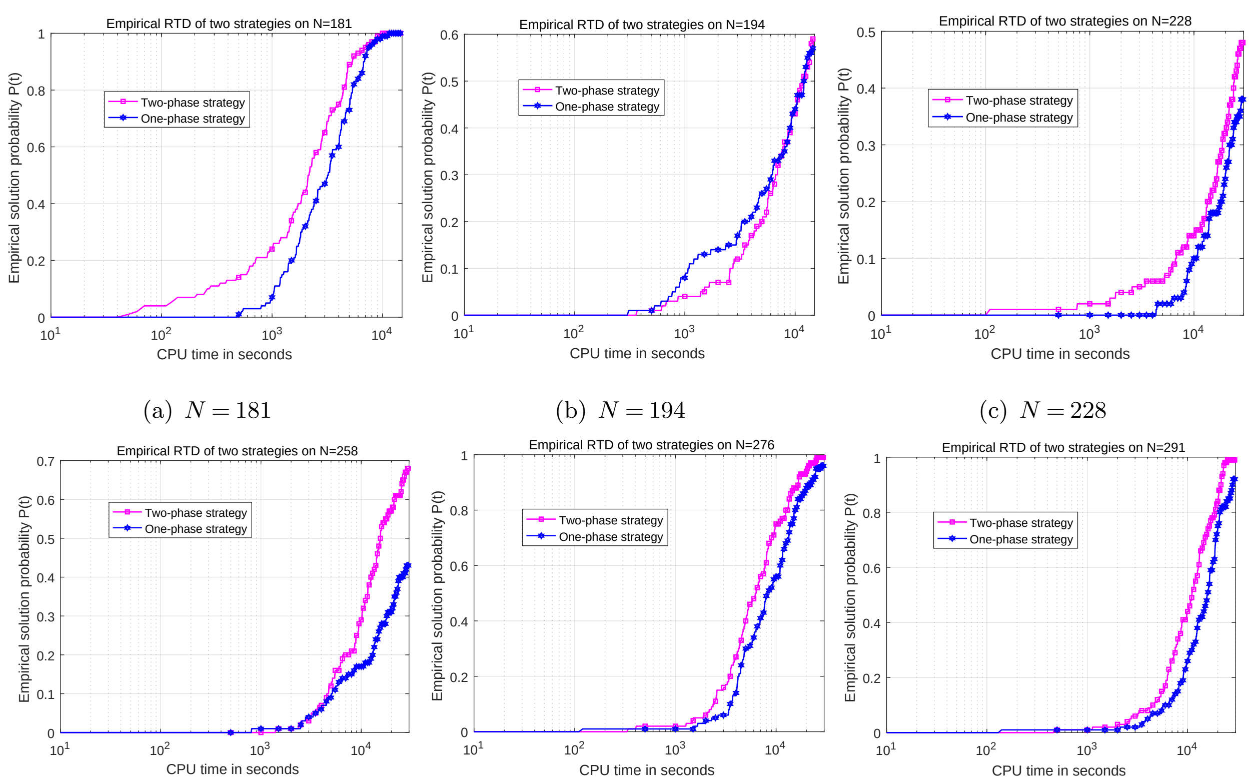
Figure 7 Empirical run-time distribution of the proposed algorithms with different perturbation strategies on the representative instances.

(2000)), the empirical RTD provides an efficient graphic representation for studying the behavior of stochastic optimization algorithms.

The experimental results are summarized in Fig. 7, which shows that the performance of each algorithm is strongly related to its perturbation operator. For example, for the PESC problem with $N = 72$, the algorithm with the SRP operator performs very well, and its success rate $P(t)$ increases rapidly to 100% as the running time t increases. However, the success rate of the algorithm with the URP operator remains 0 as the running time increases, showing that the SRP operator significantly outperforms the URP operator on this instance and plays a key role in finding high-quality solutions. On the contrary, for the PECS problem with $N = 194$, the URP operator demonstrates a high performance, while the SRP operator results in a constant success rate of 0, indicating that the URP operator appreciably outperforms the SRP operator on this instance. This phenomenon whereby the URP and SRP operators are in effect complementary is the fundamental motivation underlying our hybrid perturbation strategy. By adaptively combining the SRP and URP operators, the hybrid perturbation strategy is able to make full use of the advantages of these operators and avoid their shortcomings, thus enhancing the robustness of the proposed algorithm.

Finally, Fig. 7 shows that the hybrid perturbation strategy performs very well on all the tested instances compared to other three perturbation strategies and enables the algorithm to find improved best configurations for some hard instances (e.g., $N = 181$ and 291) with a success rate of 100%.

6.2. Effect of the two-phase search strategy



(a) $N = 181$ (b) $N = 194$ (c) $N = 228$

(d) $N = 258$ (e) $N = 276$ (f) $N = 291$

Figure 8 Comparisons between the two-phase and one-phase strategies on the representative instances.

The two-phase search strategy is another important feature of the proposed algorithm, where the first phase of algorithm undertakes to find a high-density packing configuration as quickly as possible (by gradually reducing the size of container) and the second phase is the main search engine whose goal is to search for the globally optimal solution by a multi-start design. To show the effect of the two-phase strategy, we carry out an experiment based on six representative instances. In this experiment, we create a one-phase variant of the algorithm by disabling its first phase (lines 4–11 of Algorithm 1), while keeping the other components unchanged. Then, we run both the two-phase algorithm and the one-phase variant 100 times to solve each instance and report their empirical run-time distributions of algorithms in Fig. 8.

Fig. 8 shows that for most instances the two-phase search algorithm has a better performance. Specifically, for 5 out of the 6 tested instances, the probability of obtaining the best-known solution within a given computational time is higher with the two-phase search strategy than with the one-phase search strategy. This experiment confirms the important role of the first phase of the algorithm, which significantly speeds up the search process.

7. Conclusions and Future Work

This paper presents a two-phase stochastic optimization algorithm for two challenging global optimization problems that have numerous applications: the Packing Equal Circles in a Square (PECS) problem and the Packing Equal Spheres in a Cube (PESC) problem. Our proposed algorithm is composed of a thresholding search method (combining local optimization and perturbation) and a container adjustment procedure. Computational results on a large number of widely studied benchmark instances show that the proposed algorithm is very efficient compared to the previous algorithms in the literature. In particular, for PECS, our algorithm improves the best-known results for 156 instances in the range of $2 \leq N \leq 400$ and matches the best-known results for 242 other instances. For PESC, our algorithm improves 66 best-known results in the range of $2 \leq N \leq 200$ and matches the best-known results for 124 instances. Additional experiments are performed to understand the influence of the search components and disclose the importance of the adaptive perturbation strategy.

For future research, three directions can be pursued. First, the algorithm can be adapted to solve other equal circle packing and equal sphere packing problems, such as packing equal circles into a larger circle (López and Beasley (2011), Mladenović et al. (2005), Stoyan et al. (2020)) or into a non-convex polygon container (Dai et al. (2021)), packing equal circles on the unit sphere (Clare and Kepert (1991)), packing equal spheres into a larger sphere (Birgin and Sobral (2008), Hifi and Yousef (2019)), and so on. The code of our algorithm that we make publicly available facilitates such applications. Second, the proposed approach could be extended to non-uniform size circle and sphere packing problems such as those studied in (Grosso et al. (2010), Pintér et al. (2018), Stoyan et al. (2020)) by jointly using the present perturbation strategies and the swap moves that exchange the positions of two different-size circles or spheres. Third, the thresholding search framework and the accompanying perturbation strategy are of a general nature. As such, they can be applied to the global optimization of any non-convex function having a first derivative.

Acknowledgments

We are grateful to the reviewers for their insightful comments and suggestions which helped us to improve the paper, Dr. Eckard Specht for publishing the best solutions found in this work at the Packomania website (`http://www.packomania.com`), and Prof. William Hager for publishing the source code of L-BFGS method at his personal website (`https://people.clas.ufl.edu/hager/software/`). This work was partially supported by the National Natural Science Foundation of China under the grant numbers 61703213 and 61933005.

References

Addis B, Locatelli M, Schoen F (2008) Disk packing in a square: a new global optimization approach. *INFORMS Journal on Computing* 20(4):516–524.

Akiyama J, Mochizuki R, Mutoh N, Nakamura G (2002) Maximin distance for n points in a unit square or a unit circle. *Japanese Conference on Discrete and Computational Geometry, pages 9–13* (Springer).

Amore P, Morales T (2022) Efficient algorithms for the dense packing of congruent circles inside a square. *Discrete & Computational Geometry* https://doi.org/10.1007/s00454–022–00425–5.

Bagattini F, Schoen F, Tigli L (2019) Clustering methods for large scale geometrical global optimization. *Optimization Methods and Software* 34(5):1099–1122.

Bierlaire M, Thémans M, Zufferey N (2010) A heuristic for nonlinear global optimization. *INFORMS Journal on Computing* 22(1):59–70.

Birgin EG, Sobral F (2008) Minimizing the object dimensions in circle and sphere packing problems. *Computers & Operations Research* 35(7):2357–2375.

Boll DW, Donovan J, Graham RL, Lubachevsky BD (2000) Improving dense packings of equal disks in a square. *The Electronic Journal of Combinatorics* 7:R46–R46.

Casado LG, García I, Szabo PG, Csendes T (1998) Packing equal circles packing in square. II. new results for up to 100 circles using the TAMSASS-PECS algorithm. *Optimization Theory: Recent Developments from Mátraháza* 207–224.

Castillo I, Kampas FJ, Pintér JD (2008) Solving circle packing problems by global optimization: numerical results and industrial applications. *European Journal of Operational Research* 191(3):786–802.

Clare B, Kepert D (1991) The optimal packing of circles on a sphere. *Journal of Mathematical Chemistry* 6(1):325–349.

Costa A (2013) Valid constraints for the point packing in a square problem. *Discrete Applied Mathematics* 161(18):2901–2909.

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

Dai Z, Xu K, Ornik M (2021) Repulsion-based p-dispersion with distance constraints in non-convex polygons. *Annals of Operations Research* 1–17.

Demaine ED, Fekete SP, Lang RJ (2010) Circle packing for origami design is hard. *arXiv preprint* arXiv:1008.1224v2.

Dimnaku A, Kincaid R, Trosset MW (2005) Approximate solutions of continuous dispersion problems. *Annals of Operations Research* 136(1):65–80.

Doye JP, Leary RH, Locatelli M, Schoen F (2004) Global optimization of Morse clusters by potential energy transformations. *INFORMS Journal on Computing* 16(4):371–379.

Drezner Z, Erkut E (1995) Solving the continuous p-dispersion problem using non-linear programming. *Journal of the Operational Research Society* 46:516–520.

Dueck G, Scheuer T (1990) Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics* 90(1):161–175.

Fiacco AV, McCormick GP (1964) Computational algorithm for the sequential unconstrained minimization technique for nonlinear programming. *Management Science* 10(4):601–617.

Gensane T (2004) Dense packings of equal spheres in a cube. *The Electronic Journal of Combinatorics* R33–R33.

Goldberg M (1970) The packing of equal circles in a square. *Mathematics Magazine* 43(1):24–30.

Goldberg M (1971) On the densest packing of equal spheres in a cube. *Mathematics Magazine* 44(4):199–208.

Graham R, Lubachevsky B (1996) Repeated patterns of dense packings of equal disks in a square. *The Electronic Journal of Combinatorics* 3(R16):2.

Grosso A, Jamali A, Locatelli M, Schoen F (2010) Solving the problem of packing equal and unequal circles in a circular container. *Journal of Global Optimization* 47(1):63–81.

Hager WW, Zhang H (2005) A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization* 16(1):170–192.

Hardin D, Saff E (2005) Minimal riesz energy point configurations for rectifiable d-dimensional manifolds. *Advances in Mathematics* 193(1):174–204.

Hifi M, Yousef L (2019) A local search-based method for sphere packing problems. *European Journal of Operational Research* 274(2):482–500.

Hoos HH, Stützle T (2000) Local search algorithms for SAT: An empirical evaluation. *Journal of Automated Reasoning* 24(4):421–481.

Huang W, Ye T (2010) Greedy vacancy search algorithm for packing equal circles in a square. *Operations Research Letters* 38(5):378–382.

Huang W, Ye T (2011) Global optimization method for finding dense packings of equal circles in a circle. *European Journal of Operational Research* 210(3):474–481.

Hujter M (1999) Some numerical problems in discrete geometry. *Computers & Mathematics with Applications* 38(9-10):175–178.

Joós A (2009) On the packing of fourteen congruent spheres in a cube. *Geometriae Dedicata* 140(1):49–80.

Kolossváry I, Bowers KJ (2010) Global optimization of additive potential energy functions: Predicting binary Lennard-Jones clusters. *Physical Review E* 82(5):056711.

Lai X, Hao JK, Xiao R, Glover F (2022a) Circlepacking Version v2022.0004 URL `http://dx.doi.org/10.5281/zenodo.7579558`, https://github.com/INFORMSJoC/2022.0004.

Lai X, Hao JK, Yue D, Lü Z, Fu ZH (2022b) Iterated dynamic thresholding search for packing equal circles into a circular container. *European Journal of Operational Research* 299(1):137–153.

Leary RH (2000) Global optimization on funneling landscapes. *Journal of Global Optimization* 18(4):367–383.

Liu DC, Nocedal J (1989) On the limited memory BFGS method for large scale optimization. *Mathematical Programming* 45(1):503–528.

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

López CO, Beasley JE (2011) A heuristic for the circle packing problem with a variety of containers. *European Journal of Operational Research* 214(3):512–525.

Maranas CD, Floudas CA, Pardalos PM (1995) New results in the packing of equal circles in a square. *Discrete Mathematics* 142(1-3):287–293.

Markót MC (2021) Improved interval methods for solving circle packing problems in the unit square. *Journal of Global Optimization* 81(3):773–803.

Markót MC, Csendes T (2005) A new verified optimization technique for the "packing circles in a unit square" problems. *SIAM Journal on Optimization* 16(1):193–219.

Martínez L, Andrade R, Birgin EG, Martínez JM (2009) PACKMOL: a package for building initial configurations for molecular dynamics simulations. *Journal of Computational Chemistry* 30(13):2157–2164.

M'Hallah R, Alkandari A (2012) Packing unit spheres into a cube using VNS. *Electronic Notes in Discrete Mathematics* 39:201–208.

Mladenović N, Plastria F, Urošević D (2005) Reformulation descent applied to circle packing problems. *Computers & Operations Research* 32(9):2419–2434.

Moscato P, Fontanari JF (1990) Stochastic versus deterministic update in simulated annealing. *Physics Letters A* 146(4):204–208.

Nurmela KJ, Östergård PR (1997) Packing up to 50 equal circles in a square. *Discrete & Computational Geometry* 18(1):111–120.

Nurmela KJ, et al. (1999) More optimal packings of equal circles in a square. *Discrete & Computational Geometry* 22(3):439–457.

Pei J, Dražić Z, Dražić M, Mladenović N, Pardalos PM (2019) Continuous variable neighborhood search (C-VNS) for solving systems of nonlinear equations. *INFORMS Journal on Computing* 31(2):235–250.

Pintér JD, Kampas FJ, Castillo I (2018) Globally optimized packings of non-uniform size spheres in R^d: a computational study. *Optimization Letters* 12(3):585–613.

Schaer J (1965) The densest packing of 9 circles in a square. *Canadian Mathematical Bulletin* 8(3):273–277.

Schaer J (1966) On the densest packing of spheres in a cube. *Canadian Mathematical Bulletin* 9(3):265–270.

Schwartz B (1970) Separating points in a square. *Journal of Recreational Mathematics* 3:195–204.

Specht E (2021) Packomania website. *http://www.packomania.com* .

Stoyan Y, Yaskov G, Romanova T, Litvinchev I, Yakovlev S, Cantú JMV (2020) Optimized packing multi-dimensional hyperspheres: A unified approach. *Mathematical Biosciences and Engineering* 17(6):6601–6630.

Szabó PG, Markót MC, Csendes T, Specht E, Casado LG, García I (2007) *New approaches to circle packing in a square: with program codes*, volume 6 of *Optimization and Its Applications* (Springer).

Szabó PG, Specht E (2007) Packing up to 200 equal circles in a square. *Models and Algorithms for Global Optimization*, volume 4 of *Optimization and Its Applications*, 141–156 (Springer).

Van Dam ER, Husslage B, Den Hertog D, Melissen H (2007) Maximin Latin hypercube designs in two dimensions. *Operations Research* 55(1):158–169.

Wales DJ, Doye JP (1997) Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A* 101(28):5111–5116.

Weaire D, Aste T (2008) *The pursuit of perfect packing* (CRC Press).

Wengerodt G (1983) Die dichteste packung von 16 kreisen in einem quadrat. *Beiträge zur Algebra und Geometrie - Contributions to Algebra and Geometry* 16:173–190.

Wengerodt G (1987) Die dichteste packung von 14 kreisen in einem quadrat. *Beiträge zur Algebra und Geometrie - Contributions to Algebra and Geometry* 25:25–46.

Wengerodt G, Kirchner K (1987) Die dichteste packung von 36 kreisen in einem quadrat. *Beiträge zur Algebra und Geometrie - Contributions to Algebra and Geometry* 25:147–160.