Computers & Operations Research 164 (2024) 106522

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

An efficient solution space exploring and descent method for packing equal spheres in a sphere

Jianrong Zhou [a], Shuo Ren [a], Kun He [a,*], Yanli Liu [b], Chu-Min Li [c]

[a] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China
[b] School of Science, Wuhan University of Science and Technology, Wuhan, 430081, China
[c] MIS, University of Picardie Jules Verne, Amiens, 80039, France

ARTICLE INFO

Keywords:
Global optimization
Heuristics
Equal sphere packing
Solution-space exploring & descent

ABSTRACT

The problem of packing equal spheres in a spherical container is a classic global optimization problem, which has attracted enormous studies in academia and found various applications in industry. This problem is computationally very challenging, and many efforts focus on small-scale instances with the number of spherical items less than 200 in the literature. In this work, we propose an efficient local search heuristic algorithm named solution space exploring and descent for solving this problem, which can quantify the solution's quality to determine the number of exploring actions and quickly discover a high-quality configuration. Besides, we propose an adaptive neighbor item maintenance method to speed up the convergence of the continuous optimization process and reduce the time consumption. Computational experiments on a large number of benchmark instances with $5 \leq n \leq 400$ spherical items show that our algorithm significantly outperforms the state-of-the-art algorithm. Specifically, our algorithm improves 274 best-known results and matches 84 best-known results out of the 396 well-known benchmark instances.

1. Introduction

The Sphere Packing Problem (SPP) is a classic optimization problem that seeks an arrangement of non-overlapping spherical items within a container or a bounded or unbounded space where the arrangement is usually required to be as dense as possible. SPP is an important topic in many research fields and has a variety of real-world applications. In physics, SPP is used to assist in analyzing the black hole spectrum in 3D quantum gravity (Hartman et al., 2019). In chemistry, a dense configuration of SPP can assist researchers in realizing the structures of chemical compounds (O'Toole and Hudson, 2011). In material science, the random sphere packing and particle packing models are adopted to analyze the structure of powders, liquids, proteins, colloidal suspensions, and porous materials (Clarke and Jónsson, 1993; Silbert et al., 2002; Aste et al., 2005; Klumov et al., 2011, 2014), where these models are used in the study of microscopic particle arrangement and the phenomena of fluid flow, electrical conductivity, stress distribution and other physical characteristics. In digital communication, SPP is a model in the study of telecommunication systems, optical communications, and classical-quantum channels (Valembois and Fossorier, 2004; Fazeli et al., 2014; Chaaban et al., 2016; Cheng et al., 2019). SPP also has an application on radiosurgical treatment planning (Wang, 1999).

Especially in mathematics, SPP is a classic and famous research topic. The well-known highest density $\rho^* = \pi/\sqrt{18} \approx 0.74048$ is proven to be obtained by the Face-Centred Cubic (FCC) or Hexagonal Close Packing (HCP) arrangements of the congruent sphere packing in unbounded space. And many efforts have been devoted to finding a tighter bound, contact number, and other mathematical features in various instances and spaces, such as n-dimensional ($n \geq 3$) Euclidean space (Cohn and Elkies, 2003; Bezdek, 2012; Viazovska, 2017; Cohn et al., 2017; Cohn, 2017), 3-dimensional non-Euclidean space (Kazakov et al., 2018), sphere packing on error-correcting codes (Leech and Sloane, 1971; Fazeli et al., 2015) or spherical codes (Cohn and Zhao, 2014).

SPP has a lot of variants, some of which require packing equal or unequal sphere items in a container with a specified geometric shape, such as cuboid container (Akeb, 2016; Hifi and Yousef, 2019; Hifi et al., 2023; Lai et al., 2023), spherical container (Liu et al., 2009; Zeng et al., 2012; Hifi and Yousef, 2018b), cylindrical container (Han et al., 2005; Mueller, 2005), and other shapes of containers (Birgin and Sobral, 2008; Labra and Onate, 2009; Stoyan and Yaskov, 2013; Stoyan et al., 2020). Besides, the hypersphere packing (i.e., high-dimensional sphere packing) (Stoyan and Yaskov, 2012) can be considered as an SPP variant. Furthermore, the 2-dimensional variant of SPP degenerates

* Corresponding author.
E-mail address: brooklet60@gmail.com (K. He).

https://doi.org/10.1016/j.cor.2023.106522
Received 13 May 2023; Received in revised form 6 December 2023; Accepted 20 December 2023
Available online 4 January 2024

into circle packing problems (He et al., 2018, 2021; Lai et al., 2022, 2023), which are well-known and popular topics in academia and industry.

One important variant of SPP is Packing Equal Spheres in a Sphere (PESS) (Huang and Yu, 2011; M'Hallah et al., 2013; Hifi et al., 2017; Hifi and Yousef, 2018b). Given n unit spherical items, PESS aims to pack the n spherical items into a spherical container with the least radius. Due to its simplicity in form, PESS is a classic and representative problem in the SPP family.

However, even the restricted models of geometric packing problems (e.g., circle packing problems) have been proven to be NP-hard (Fowler et al., 1981; Demaine et al., 2016). Thus, solving SPP, which can be regarded as an extension of circle packing problems, is computationally very challenging. Most works aim to design an algorithm for solving SPP on small-scale instances where the number of packing items is less than 200 ($n \leq 200$). Only a few studies have attempted to solve SPP on a large scale, and the algorithms proposed in these studies require a significant amount of time to obtain a dense configuration of SPP.

In this work, we address the PESS variant of SPP on small and moderate scales with the number of packing items up to 400. We employ the elastic model (also known as the Quasi-Physical Quasi-Human model, QPQH) (Huang and Xu, 1999; Huang and Yu, 2011) of PESS. The intersection of item-item and item-container is allowed in this model, and an elastic objective function is defined to evaluate the degree of intersection. Then, the intersection can be reduced or eliminated by adopting a continuous optimization algorithm to minimize the elastic objective function, such as the gradient descent method (Huang and Yu, 2011) or the quasi-Newton method (He et al., 2018; Lai et al., 2022, 2023).

Based on the PESS elastic system, we propose an advanced local search heuristic, termed Solution-space Exploring and Descent (SED), to solve the PESS problem. SED iteratively performs two processes, "Exploring" and "Descent", to improve the best-found solution during the iterative heuristic search. In the "Exploring" process, SED perturbs the operated solution to obtain a set of candidate solutions, and the number of perturbations depends on a merit function, which can quantify the operated solution's quality. In the "Descent" process, SED employs a strategy to select a high-quality solution from the set of candidate solutions and set it as the operated solution in the next iteration. Experiments show that SED could efficiently find a dense configuration by executing the iteration process on small and moderate instances.

In addition, we propose an Adaptive Neighbor item Maintenance (ANM) method to maintain the neighbor structure (He et al., 2018) for the PESS elastic system. The neighbor structure stores adjacent item messages for each packing item to reduce the time consumption of calculating the elastic objective function and its gradient. ANM uses two variables "counter" and "deferring length" to achieve the adaptive feature, which maintains the neighbor structure in the continuous optimization process. ANM reconstructs the neighbor structure at each iteration when the layout changes significantly. Otherwise, ANM defers the neighbor structure maintenance. In this way, ANM can reduce the number of times to reconstruct the neighbor structure and accelerate the continuous optimization process of PESS.

Extensive experimental results show that our proposed algorithm significantly outperforms the state-of-the-art algorithm. Specifically, our algorithm yields 14 better, 30 equal, and 2 worse results than the state-of-the-art algorithm out of the 46 comparing instances ($5 \leq n \leq 50$). In addition, our algorithm improves the best-known results for 274 instances, matches the best-known results for 84 instances, and obtains worse results for 38 instances out of the 396 benchmark instances. Besides, further analyses show that our proposed ANM module can defer over 50% unnecessary maintenance and reduce over 30% time consumption in the continuous optimization process of PESS.

The main contributions of this work are summarized as follows:

- We propose an efficient Solution space Exploring and Descent (SED) heuristic for solving the PESS problem.
- We propose an Adaptive Neighbor item Maintenance (ANM) method of maintaining the neighbor structure for solving the PESS problem, which is a general method and can be easily adapted for other packing problems.
- Extensive experiments on a large number of benchmark instances with up to $n = 400$ demonstrate our proposed algorithm's excellent performance and efficiency, gaining new improved configurations on many instances.

The rest of this paper is organized as follows. Section 2 presents the related works of the SPP and PESS problem, including some construction methods, typical models, heuristics and metaheuristics, and recent works for solving the PESS problem. Section 3 introduces the mathematical formula of the PESS problem and the classic elastic model (QPQH) for solving PESS. Section 4 presents the main framework of our algorithm and other components, including the initialization, SED heuristic, container adjustment approach, neighbor structure, ANM module, and the continuous optimization algorithm of PESS. Section 5 presents the experimental results of our proposed algorithm compared with the state-of-the-art algorithm and best-known results, parameter study, and the analysis of the neighbor structure and the ANM module. The conclusion is drawn in the end.

2. Related work

There are two main categories of algorithms for solving the SPP, namely random sphere packing and dense sphere packing.

Algorithms of random sphere packing are designed for specified purposes, such as various container filling, particle microstructure analysis, physical characteristic analysis, etc. The main idea of these algorithms belongs to the simulation method (Silbert et al., 2002; Wouterse and Philipse, 2006; Shi and Zhang, 2008; Liu et al., 2017) or construction method (Han et al., 2005; Soontrapa and Chen, 2013; Chen and Zhao, 2022). Due to the characteristics of these algorithms, they can easily obtain a configuration with a number of packing spheres up to hundreds or thousands, but the packing density is not very high.

Algorithms of dense sphere packing aim to solve SPP as an optimization problem, with the goal of finding a configuration that is as dense as possible, packing as many items as possible into a specified container, and using as few bins as possible (in the case of bin packing problems), among other objectives. Our literature review focuses on the optimization version of dense SPP, and the problem of packing spheres into a cuboid or spherical container is the most popular and representative in the SPP family.

Hifi and Yousef (2014) propose a width-beam search heuristic to pack the spheres one by one into the container to find a feasible solution, and a hill-climbing strategy is proposed for improving the width-beam heuristic. Meanwhile, A dichotomous search is adopted to find a dense configuration. It is followed by several researches (Hifi and Yousef, 2015, 2016, 2018a) based on similar ideas that improve the algorithm performance.

Hifi and Yousef (2019) propose an efficient local search method with multi-strategies, including using a basic greedy local strategy to ensure a feasible solution, employing the elastic model (QPQH) to solve the decision problem of SPP, and a drop and rebuild method for perturbation. Akeb (2016) proposes a multi-level look-ahead strategy and some population-based algorithms (Hifi et al., 2022, 2023) are proposed for solving this problem.

Huang and Yu (2011) present the classic and powerful elastic model (QPQH) for solving SPP, and a serial symmetrical relocation strategy is proposed for perturbation. Zeng et al. (2012) extend this model to solve the unequal spheres packing problem. Liu et al. (2009) propose an energy landscape paving method and combine it with the gradient descent method based on the elastic model to solve the circle and sphere packing problems. Hifi et al. (2017) propose an adaptive particle swarm optimization algorithm based on the elastic model to solve PESS, and an extension work (Hifi and Yousef, 2018b) is presented for other equal sphere packing problems. M'Hallah and Alkandari (2012), M'Hallah et al. (2013) apply the non-linear program and variable neighborhood search method to solve the equal sphere packing problem. There are also some works based on the non-linear programming method to solve the sphere packing problem in various containers (Birgin and Sobral, 2008) and hyperspaces (Stoyan and Yaskov, 2012; Stoyan et al., 2020).

The well-known Packomania website (Specht, 2023) maintained by Specht presents many circle and sphere packing problems and records their best-known solutions. From the recently updated history on the PESS problem at Packomania, the best-known solutions for $1 \leq n \leq 25$ are provided by mathematical analysis except for $n = 24$, Huang and Yu (2011) hold several best-known solutions for $26 \leq n \leq 160$, and Specht holds the remaining best-known solutions with unpublished methods.

In summary, most efforts are based on local search, heuristic, meta-heuristic, and non-linear programming methods to solve the optimization version of SPP. The elastic model (QPQH) based methods (Huang and Yu, 2011; Hifi and Yousef, 2018b) can be regarded as the state-of-the-art algorithms for solving the PESS problem.

3. Preliminaries

3.1. Problem formulation

The PESS problem aims to pack n unit spheres $\{s_1, s_2, \ldots, s_n\}$ into a spherical container and minimizes the container radius while subjected to two constraints: (I) No pair of unit spheres overlap with each other; (II) No unit sphere exceeds the spherical container. The PESS problem can be formulated in three-dimensional Cartesian coordinate system as a non-linear constrained optimization problem:

$$\text{Minimize} \quad R$$

$$s.t. \quad \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \geq 2, \quad 1 \leq i, j \leq n, \ i \neq j, \tag{1}$$

$$\sqrt{x_i^2 + y_i^2 + z_i^2} + 1 \leq R, \quad 1 \leq i \leq n, \tag{2}$$

where R is the radius of the spherical container centered at the origin $(0, 0)$, and the center of the unit sphere s_i is located at (x_i, y_i, z_i). Eqs. (1) and (2) correspond to the two constraints (I) and (II), respectively.

3.2. The elastic model for PESS

Given a fixed spherical container of radius R, it is difficult to find a feasible solution or determine whether there exists a feasible solution. To cope with this difficulty, the elastic model (Huang and Xu, 1999; Huang and Yu, 2011), which can be regarded as a relaxation of PESS, is proposed for solving the PESS problem. The overlap of sphere-sphere and sphere-container is allowed in this model, and an objective function called elastic energy E is designed to quantify the overlapping degree of a candidate solution. The goal of the algorithm based on this model is to minimize the elastic energy E, which is equivalent to minimizing the overlapping degree. In this way, the elastic model converts the PESS problem to an unconstrained and non-convex continuous optimization problem.

Definition 3.1 (*Overlapping Degree*)**.** The overlapping degree of two unit spheres s_i and s_j, denoted as O_{ij}, is defined as follows:

$$O_{ij} = \max\left(0,\ 2 - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}\right), \tag{3}$$

And the overlapping degree of a unit sphere s_i to the spherical container, denoted as O_{i0}, is defined as follows:

$$O_{i0} = \max\left(0,\ \sqrt{x_i^2 + y_i^2 + z_i^2} + 1 - R\right). \tag{4}$$

Fig. 1 shows a conflicting example with the two types of overlaps. According to the definition of elastic potential energy in physics, the square of the embedding degree is proportional to the elastic potential energy, giving the following elastic energy definition.

Definition 3.2 (*Elastic Energy*)**.** The total elastic energy E of the PESS system is defined as follows:

$$E_R(\boldsymbol{x}) = E(\boldsymbol{x}, R) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} O_{ij}^2 + \sum_{i=1}^{n} O_{i0}^2, \tag{5}$$

where R is the container radius, and $\boldsymbol{x} = [x_1, y_1, z_1, x_2, y_2, z_2, \ldots, x_n, y_n, z_n]^{\mathrm{T}}$ is a vector, $\boldsymbol{x} \in \mathbb{R}^{3n}$, representing a candidate solution.

Note that the elastic energy $E_R(\boldsymbol{x})$ quantifies the overlapping degree of a candidate solution $\boldsymbol{x}$. If the condition $E_R(\boldsymbol{x}) = 0$ is met, then there is no overlap in the corresponding solution $\boldsymbol{x}$ (i.e., Eqs. (1) and (2) are satisfied), indicating that $\boldsymbol{x}$ is a feasible solution for the PESS problem. Since we solve the problem with the fixed container radius R and use a sequential unconstrained optimization approach to make the container adjustment, we omit the subscript R in $E_R(\boldsymbol{x})$ throughout the rest of the paper and simply denote it as $E(\boldsymbol{x})$ for the sake of readability.

4. The proposed algorithm for PESS

In this section, we introduce the main framework of our algorithms, including the Solution-space Exploring and Descent (SED) heuristic, the Adaptive Neighbor item Maintenance (ANM) method, and other minor components.

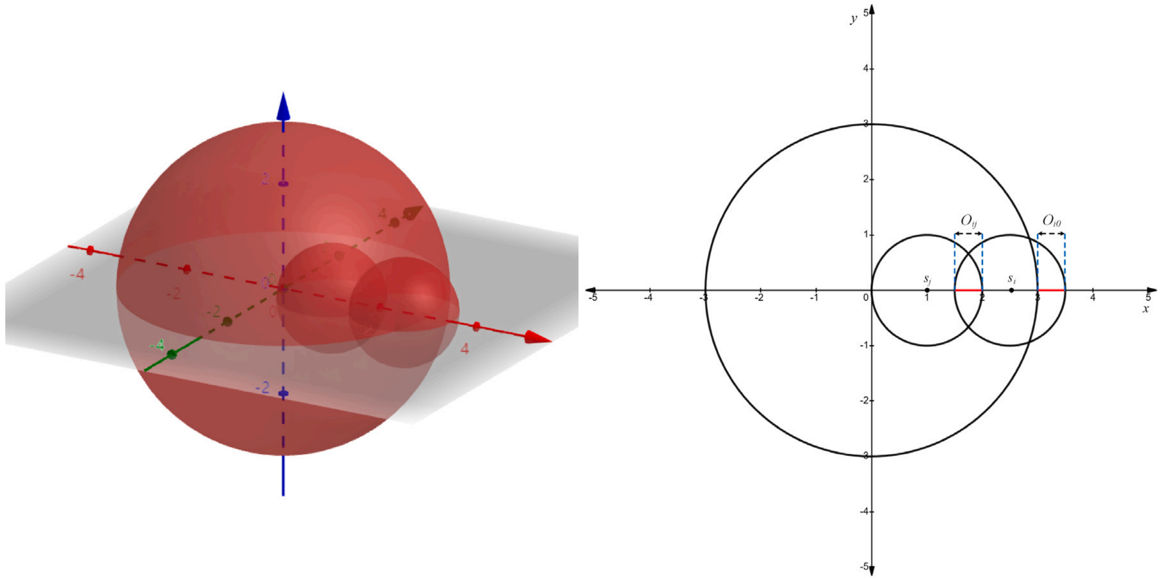
4.1. Main framework

We first introduce our algorithm framework, of which the pseudocode is depicted in Algorithm 1. The algorithm first obtains an initial solution (line 1) by employing the initialization procedure, described in Section 4.2, and the initial solution is recorded as the current best solution $(\boldsymbol{x}^*, R^*)$ (line 2). Subsequently, the algorithm performs an iterative search process to improve the current best solution until the cut-off time T_{cut} is reached (lines 3-10).

In the iterative search process, the current radius is set as the best radius found so far (line 4). The algorithm employs the SED heuristic, described in Section 4.3, to obtain a feasible solution or an infeasible solution with the smallest energy (i.e., the smallest overlapping degree) (line 5). Then, the container radius adjustment approach is employed to adjust the packing spheres' position and the container radius until the overlaps are eliminated and a feasible solution with a minimal container radius (line 6) is obtained. After that, the current best solution is updated when a better solution is found (lines 7–9). Finally, the

Algorithm 1: The framework for solving PESS

```
1 Input: A number of unit spheres n; A cut-off time T_cut
2 Output: A feasible solution with the minimal container radius
    (x*, R*)
 1: (x, R) ← initialize(n)
 2: x* ← x,   R* ← R
 3: while time() ≤ T_cut do
 4:     R ← R*
 5:     x ← SED(n, R)
 6:     (x, R) ← adjust_container(x, R)
 7:     if R < R* then
 8:         x* ← x,   R* ← R
 9:     end if
10: end while
11: return (x*, R*)
```



(a) Three-dimensional view (b) Sectional view for $z = 0$

Fig. 1. Illustration of an example with the two types of overlaps where the center of the container is located at the origin $(0,0)$ with $R = 3$ and two spherical items s_i and s_j are located at $(2.5, 0, 0)$ and $(1, 0, 0)$, respectively, with $r = 1$. (a) shows the three-dimensional illustration, and (b) shows the sectional view for $z = 0$ where O_{ij} and O_{i0} indicate the overlaps of sphere-sphere and sphere-container, respectively.

algorithm returns the solution $(\boldsymbol{x}^*, R^*)$ as the final configuration (line 11).

4.2. Initialization

The initialization module of the algorithm aims to rapidly provide a high-quality feasible solution to serve as a good starting point for the iterative search process.

The search algorithm based on the elastic model for solving the PESS problem has a limitation in needing a fixed container radius. Thus, estimating a lower bound radius is necessary for the search process. Based on the density equation, one way to obtain a lower bound radius is as follows:

$$\rho = \frac{nV_{sph.}}{V_{con.}}, \quad V_{sph.} = \frac{4}{3}\pi r^3, \quad V_{con.} = \frac{4}{3}\pi R^3, \tag{6}$$
$$\Longrightarrow \rho = \frac{nr^3}{R^3},$$
$$\Longrightarrow R = \sqrt[3]{\frac{n}{\rho}}, \quad (r = 1).$$

where n is the number of packing spheres, r and R represent the radius of packing spheres and the spherical container in which $r = 1$ corresponds to the PESS problem. $V_{sph.}$ and $V_{con.}$ indicate the volume of packing spheres and the spherical container, respectively, and ρ denotes the density that is the ratio of the total volume of packing spheres to the volume of the spherical container.

The highest density $\rho \simeq 0.74048$ is proved amongst all possible spherical lattice packings. However, the lower bound based on this density is too tight for the PESS problem on small and moderate scales. Therefore, we reference the density of the best-known results of PESS on the Packomania website (Specht, 2023) for $n \le 400$ and empirically set $\rho = 0.6$ so as to estimate the lower bound of the radius R in this work, so that, the pseudocode of the initialization module depicted in Algorithm 2 uses the density of $\rho = 0.6$ to estimate a lower bound radius that is $R = \sqrt[3]{\frac{n}{0.6}}$.

Then, SED is employed with the fixed radius R to find a candidate solution with the smallest energy. Following that, the container adjustment approach is applied to obtain a feasible solution with the smallest container radius $(\boldsymbol{x}, R)$, and it is returned as a good start for the iterative search process.

Algorithm 2: Initialize (n)

Input: A number of unit spheres n
Output: A feasible solution with the minimal container radius $(\boldsymbol{x}, R)$
1: $R \leftarrow \sqrt[3]{\frac{n}{0.6}}$
2: $\boldsymbol{x} \leftarrow \mathrm{SED}(n, R)$
3: $(\boldsymbol{x}, R) \leftarrow \mathrm{adjust\_container}(\boldsymbol{x}, R)$
4: **return** $(\boldsymbol{x}, R)$

4.3. SED heuristic

Given that we adopt the elastic model, our proposed SED heuristic aims to solve the following problem: given a fixed container radius, the goal is to find a feasible solution for the PESS problem. If a feasible solution is found, the algorithm returns it immediately. Otherwise, the algorithm returns an infeasible solution with the smallest energy during the search process. It can be regarded as a decision problem of the PESS problem.

To design an efficient local search heuristic, the key is to design an efficient solution exploration strategy. Starting from an initial low-quality solution, the algorithm is expected to find a high-quality solution by performing explorations and iterations as relatively few as possible. Given that a feasible solution is more potentially located in the neighborhood of the high-quality solution in the solution space than the low-quality one, *it is worth making more exploration actions based on high-quality solutions to discover a feasible solution*. According to this idea, we define a new merit function $J(\boldsymbol{x})$, formulated as follows:

$$J(\boldsymbol{x}) = \lceil -c \log_2 E(\boldsymbol{x}) \rceil, \tag{7}$$

where function J is the ceiling of the negative logarithm of the energy E, and c is a coefficient that controls the value of function J. The function J maps the energy E to an integer, which is applied to control the exploration number in the heuristic search process where a low-

Algorithm 3: SED(n, R)

1 **Input**: A number of unit spheres n; A container radius R
2 **Output**: A feasible solution or an infeasible solution with the smallest energy found so far $\boldsymbol{x}^*$

1: $\boldsymbol{x} \leftarrow$ random_layout(n, R)
2: $\boldsymbol{x} \leftarrow$ L-BFGS-ANM($E, \boldsymbol{x}$)
3: $i \leftarrow 0, \quad \boldsymbol{x}^* \leftarrow \boldsymbol{x}$
4: **while** $i < S_{iter}$ **and** time() $\leq T_{cut}$ **do**
5:   **if** $E(\boldsymbol{x}^*) \leq \epsilon$ **then**
6:     **break**
7:   **end if**
8:   $m \leftarrow \max(1, J(\boldsymbol{x})), \quad C \leftarrow \emptyset$
9:   **for** j from 1 to m **do**
10:     $\boldsymbol{x}' \leftarrow$ perturbing($\boldsymbol{x}$)
11:     $\boldsymbol{x}' \leftarrow$ L-BFGS-ANM($E, \boldsymbol{x}'$)
12:     $C \leftarrow C \cup \{\boldsymbol{x}'\}$
13:   **end for**
14:   $\boldsymbol{x} \leftarrow$ select($C, \boldsymbol{x}$)
15:   **if** $E(\boldsymbol{x}) < E(\boldsymbol{x}^*)$ **then**
16:     $\boldsymbol{x}^* \leftarrow \boldsymbol{x}$
17:   **end if**
18:   $i \leftarrow i + 1$
19: **end while**
20: **return** $\boldsymbol{x}^*$

quality solution with larger energy is assigned to a small exploration number and a high-quality solution with smaller energy is assigned to a large exploration number. The coefficient is set to $c = 7$ as default.

The pseudocode of SED is depicted in Algorithm 3. The call of SED(n, R) returns a feasible solution or an infeasible solution with the smallest energy if the feasible solution cannot be found. Initially, SED generates a random layout as the initial solution $\boldsymbol{x}$ where the unit spheres are randomly packed into the container with possible overlaps (line 1). Then, SED employs the L-BFGS-ANM optimizer to minimize the energy of the initial solution $\boldsymbol{x}$ (line 2) of which L-BFGS-ANM is the classic Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (Liu and Nocedal, 1989) combined with the ANM module (discussed in Section 4.5), and the initial solution is recorded as the current best solution $\boldsymbol{x}^*$ (line 3). Subsequently, SED performs several iterative search processes to improve the best solution $\boldsymbol{x}^*$.

At each iterative search process, SED uses the current operated solution $\boldsymbol{x}$ to calculate the exploration number m by Eq. (7) (line 8). Then SED perturbs the current operated solution $\boldsymbol{x}$ to obtain a perturbed solution $\boldsymbol{x}'$ and employs L-BFGS-ANM to minimize the energy of the perturbed solution $\boldsymbol{x}'$ (lines 10–11). SED repeats this perturbing operation for m times until a perturbed solution set C, $|C| = m$, is obtained (lines 9–13). After that, SED adopts a "select" operator to choose a candidate solution from the perturbed solution set C as the offspring solution $\boldsymbol{x}$ (line 14) and operates it in the next iteration. The current best solution $\boldsymbol{x}^*$ is updated if a better solution $\boldsymbol{x}$ is found, i.e., $E(\boldsymbol{x}) < E(\boldsymbol{x}^*)$ (lines 15–17). If a feasible solution is found, SED returns the solution immediately (lines 5–7) where the solution $\boldsymbol{x}^*$ is regarded as a feasible solution when $E(\boldsymbol{x}^*)$ is tiny enough ($E(\boldsymbol{x}^*) \leq \epsilon$ and ϵ is set as 10^{-25} in this work). Otherwise, SED returns the best solution $\boldsymbol{x}^*$ with the smallest energy as the result when reaching the maximum number of iteration steps S_{iter} or exceeding the time limit ($S_{iter} = 700$ is set as default).

To obtain a perturbed solution $\boldsymbol{x}'$ (line 11), we randomly shift the coordinate of packed unit spheres in the operated solution $\boldsymbol{x}$, and it can be described as follows, $x_i' \leftarrow x_i + r_x$, $y_i' \leftarrow y_i + r_y$, $z_i' \leftarrow z_i + r_y$ $(1 \leq i \leq n)$ where r_x, r_y, r_z are random numbers, $r_x, r_y, r_z \in U(-\theta, \theta)$, U stands for uniform distribution and $\theta = 0.8$ is set as default.

The strategy of the "select" operator is described as follows:

$$\text{select}(C, \boldsymbol{x}) = \begin{cases} \arg\min\limits_{\boldsymbol{x}' \in C} \ E(\boldsymbol{x}'), & \text{if } \min\limits_{\boldsymbol{x}' \in C} E(\boldsymbol{x}') < E(\boldsymbol{x}) \\ P\left(X = \boldsymbol{x}' \mid p_{\boldsymbol{x}'} = \text{softmax}(J(\boldsymbol{x}'))\right), & \text{otherwise} \end{cases}$$

and the "softmax" function is defined as follows:

$$\text{softmax}(J(\boldsymbol{x}')) = \frac{\exp(J(\boldsymbol{x}'))}{\sum_{\boldsymbol{x}'' \in C} \exp(J(\boldsymbol{x}''))}.$$

The "select" operator first compares the current operated solution $\boldsymbol{x}$ with the candidate solution $\boldsymbol{x}'$ with the smallest energy in C, and the solution $\boldsymbol{x}$ is replaced by $\boldsymbol{x}'$ if the condition $E(\boldsymbol{x}') < E(\boldsymbol{x})$ is met. Otherwise, the operator employs a "softmax" function to choose a candidate solution in C to replace the solution $\boldsymbol{x}$.

4.4. Container adjustment approach

Assuming we obtain a solution by employing the SED heuristic, the solution may be feasible but probably contain overlaps, thus being infeasible. Now, we aim to solve another problem described as follows. (I) If the solution is feasible, the problem is adjusting the packing spheres' position and shrinking the container to obtain a better solution. (II) If the solution is infeasible, the problem is adjusting the packing spheres' position and expanding the container until the overlap is eliminated to obtain a feasible solution with a minimal container radius.

The most intuitive and popular method to deal with this problem is the binary search approach (Huang and Yu, 2011; He et al., 2018). Inspired by the packing equal circles in a circle problem, we adopt a smart and significantly faster method (Lai et al., 2022, 2023) to solve the PESS problem, which is presented as follows.

Let vector $\boldsymbol{z} = [x_1, y_1, z_1, x_2, y_2, z_2, \ldots, x_n, y_n, z_n, R]^{\mathrm{T}}$, $\boldsymbol{z} \in \mathbb{R}^{3n+1}$, be a candidate solution with the container radius R being a variable. And a new objective function U can be reformulated as follows:

$$U_\lambda(\boldsymbol{z}) = U(\boldsymbol{z}, \lambda) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} O_{ij} + \sum_{i=1}^{n} O_{i0} + \lambda R^2,$$

where O_{ij} and O_{i0} is defined in Eqs. (3) and (4), λR^2 is a penalty term and λ is a penalty coefficient.

The problem mentioned above is converted into a series of unconstrained optimization problems. A feasible solution with a minimal container radius can be obtained by consecutively solving the problem of minimizing the objective function U while decreasing the λ value, starting from a candidate solution and an initial λ value.

Now, we introduce the container adjustment approach, and its pseudocode is presented in Algorithm 4. Given a candidate solution $\boldsymbol{x}$ and a container radius R. The algorithm starts from an initial λ value (empirically set to 10^{-4} in this work) and performs several iterations to obtain a feasible solution with a minimal container radius. At each iteration, the algorithm employs L-BFGS-ANM to minimize the objective function $U_\lambda(\boldsymbol{z})$ and updates the solution $\boldsymbol{z}$ in which $\boldsymbol{z}$ consists of the candidate solution $\boldsymbol{x}$ and the container radius R. Then, the coefficient λ is halved and the algorithm continually adjusts the solution $\boldsymbol{z}$ in the next iteration. After several iterations, the objective function $U_\lambda(\boldsymbol{z})$ converges to 0 so that the overlaps are tiny enough in the solution $\boldsymbol{z}$, which is decomposed into $(\boldsymbol{x}^*, R^*)$ returned as the result.

It is worth noticing that the penalty term λR^2 is tiny enough after sufficient iterations. The objective function U degenerates to the energy E (Eq. (5)), and the algorithm forces to minimize the energy E (i.e., eliminate overlaps) without the fixed radius constraint to obtain a feasible solution.

4.5. Adaptive neighbor item maintenance

In this subsection, we first introduce the efficient neighbor structure for PESS, then present our proposed ANM module to maintain the

Algorithm 4: adjust_container($\boldsymbol{x}, R$)

Input: A candidate solution $\boldsymbol{x}$; A container radius R
Output: A feasible solution with a minimal container radius $(\boldsymbol{x}^*, R^*)$

```
z ← (x, R),   λ ← 10^-4
for i from 1 to 35 do
    z ← L-BFGS-ANM(U_λ, z)
    λ ← 0.5 × λ
end for
(x*, R*) ← z
return (x*, R*)
```

neighbor structure adaptively and the L-BFGS-ANM optimizer for the continuous optimization process of PESS.

An efficient neighbor structure (He et al., 2018; Lai et al., 2022, 2023) has been proposed to solve the problem of packing equal circles in a circle. This structure can significantly reduce the time complexity of calculating the energy E and its gradient (from $O(n^2)$ to $O(n)$ in two-dimensional packing problems (He et al., 2018)) and accelerate the convergence of the continuous optimization process. We adopt this structure to solve the PESS problem and introduce it as follows.

Let l_{ij} indicates the Euclidean distance between the centers of two unit spheres s_i and s_j, given by the following equation:

$$l_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}.$$

Recall that O_{ij} denotes the overlapping distance between two unit spheres s_i and s_j defined by Eq. (3). It is clear that $O_{ij} > 0$ when $l_{ij} < 2$, and $O_{ij} = 0$ otherwise. And we define the neighbor $\Gamma(i)$ of unit sphere s_i to be a subset of n unit spheres $\{s_1, s_2, \dots, s_n\}$ as follows:

$$\Gamma(i) = \left\{ s_j \mid \forall j : 1 \le j \le n,\ i \ne j,\ l_{ij} < l_{cut} \right\},$$

where l_{cut} is a distance controlling parameter. If l_{cut} is set to 2, then all the unit spheres s_j overlapping with unit sphere s_i are contained in the neighbor $\Gamma(i)$. Therefore, the energy concerning sphere s_i can be calculated by enumerating the spheres in $\Gamma(i)$ instead of enumerating all n unit spheres, and the energy $E(\boldsymbol{x})$ (Eq. (5)) can be reformulated as follows:

$$E(\boldsymbol{x}) = \sum_{i=1}^{n} \sum_{s_j \in \Gamma(i)} O_{ij}^2 [i < j] + \sum_{i=1}^{n} O_{i0}^2,$$

where "[]" is the Iverson bracket that $[P] = 1$ if statement P holds, otherwise $[P] = 0$. The statement "$i < j$" ensures the overlaps of pairwise spheres are only calculated once.

We empirically set the default value of l_{cut} to 4 in this work, as setting it too large can result in the neighbor set containing too many non-adjacent packed spheres, increasing the time cost of the energy and gradient computation. Conversely, if l_{cut} is set too small, the correctness of the energy and gradient calculation cannot be ensured without maintaining the neighbor set when the packing spheres have minor shifts. Fig. 2 gives an example to show the neighbor of a packing item on equal circle packing problems with three different l_{cut} settings. Several approaches can be applied to construct the neighbor structure, such as brute force (enumerating all pairwise spheres), scan line, and k-d tree, etc. The k-d tree algorithm is a very efficient and general approach to handling high-dimensional problems, so we adopt the k-d tree algorithm to construct the neighbor structure in this work.

Now, we introduce our proposed ANM module. The ANM module is incorporated in the L-BFGS algorithm, denoted as L-BFGS-ANM, and the pseudocode of L-BFGS-ANM is depicted in Algorithm 5. The call of L-BFGS-ANM($f, \boldsymbol{x}$) returns a variable $\boldsymbol{x}^*$ with a local minimum of the objective function $f(\boldsymbol{x}^*)$. To accomplish the adaptive maintenance feature, ANM maintains two variables during the continuous optimization program: the deferring counter cnt and the deferring length len.

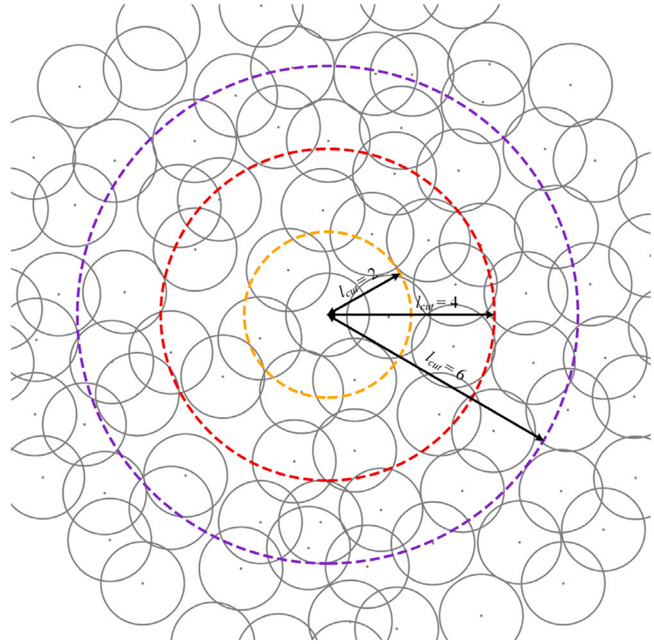


Fig. 2. Illustration of the neighbor on the equal circle packing problem with different l_{cut} settings. This illustrative example gives three settings for l_{cut} = 2, 4, 6 on a conflicting layout. We empirically set l_{cut} = 4 as a trade-off setting for the PESS problem in this work.

Algorithm 5: L-BFGS-ANM ($f, \boldsymbol{x}$)

Input: An objective function f; A variable $\boldsymbol{x}$
Output: A variable $\boldsymbol{x}^*$ with the value $f(\boldsymbol{x}^*)$ is a local minimum

```
1:  cnt ← 0,   len ← 1
2:  construct a current neighbor Γ
3:  for k from 0 to MaxIter do
4:      d_k ← two_loop_recursion()
5:      α_k ← arg min_{α∈ℝ} f(x + α d_k)
6:      x ← x + α_k d_k
7:      cnt ← cnt + 1
8:      if cnt ≥ len then
9:          construct a new neighbor Γ'
10:         if Γ ≠ Γ' then
11:             cnt ← 0,   len ← 1,   Γ ← Γ'
12:         else
13:             cnt ← 0,   len ← 2 × len
14:         end if
15:     end if
16:     if ||g(x)||_2 ≤ 10^-12 then
17:         break
18:     end if
19: end for
20: x* ← x
21: return x*
```

At the beginning of the L-BFGS-ANM procedure, the counter cnt and length len are initialized to 0 and 1, respectively, and a neighbor set Γ is constructed (lines 1–2). Then, L-BFGS-ANM uses the neighbor to calculate the energy and gradient to iteratively update the variable $\boldsymbol{x}$ (lines 3–19) and finally obtains a variable $\boldsymbol{x}^*$ at a local minimum.

At each iteration, L-BFGS-ANM employs the classic two-loop recursion approach, which is the core of L-BFGS, to calculate the current descent direction d_k (line 4). Then, a line search approach is applied to obtain a step length α_k (line 5), and the variable $\boldsymbol{x}$ is updated (line 6). Followed by the main part of the ANM module (lines 7–15), the counter cnt is incremented by 1 and compared with the deferring length len. The neighbor maintenance process is triggered if the counter cnt reaches the deferring length len.

When the maintenance process is triggered, a new neighbor structure Γ' is constructed and compared with the current neighbor structure Γ. If two neighbor structures Γ and Γ' are different, the current neighbor structure is updated by Γ', and the counter cnt and deferring length are reset to 0 and 1, respectively. Otherwise, the counter cnt is reset to 0, and the deferring length len is multiplied by 2.

At the end of the iteration, the algorithm checks the norm of the gradient, and the iterative process is terminated if the norm is tiny enough (lines 16–18) or the maximum iteration step is reached. Finally, the objective function $f(\boldsymbol{x})$ can be regarded as reaching a local minimum, and the corresponding variable $\boldsymbol{x}$ is returned as the final result.

The mechanism of the ANM module can be described as follows. If the layout is unstable, the neighbors obtained at each iteration are different, which causes the counter cnt and the deferring length len to be constantly reset to 0 and 1, respectively, and the maintenance process is triggered when the ANM module is called. Otherwise, the length len grows exponentially to defer the maintenance process. Note that ANM is a general adaptive method, which can be adopted for other problems like dynamic packing problems and online packing problems (Ye et al., 2009, 2011; Brubach, 2015; Hokama et al., 2016; Fekete et al., 2019; Lintzmayer et al., 2019; Epstein, 2019) based on the elastic model or other continuous optimization algorithms.

Some works also employ the neighbor structure to solve packing problems, and they also propose some maintenance strategies. He et al. (2018) use a simple method to maintain the neighbor, consisting in reconstructing the neighbor every 10 iterations. Reconstructing the neighbor is unnecessary when the layout is stable (i.e., the relative position of each packing sphere does not change). In this case, this method will waste computational resources. Lai et al. (2022) propose a two-phase strategy to maintain the neighbor. In the first phase, it calculates the energy and gradient by enumerating all the pairwise circles without using the neighbor structure. In the second phase, it constructs a neighbor at the beginning and uses the neighbor to calculate the energy and gradient without updating until the end. This strategy has several disadvantages: (1) The enumeration method in the first phase is computationally expensive; (2) The changing condition from the first phase to the second phase requires particular expert experience and needs to be fine-tuned in different problems; (3) The neighbor is not updated in the second phase so that if the solution falls into a saddle point, the correctness of this strategy cannot be guaranteed. In contrast, our ANM method can effectively address these issues.

5. Experiments and discussions

In this section, we first present our experimental setup. Then, we compare our algorithm with the state-of-the-art PSO-BA method (Hifi and Yousef, 2018b) , evaluate our algorithm's performance on a large number of benchmark instances, and make a comparison with the best-known results from Packomania (Specht, 2023) (downloaded on March 1st, 2023). Finally, we present the parameter study and the performance analysis of the neighbor structure and the ANM module.

5.1. Experimental setup

Our algorithm was implemented in the C++ programming language and compiled using g++ 5.4.0. Experiments were performed on a server with Intel® Xeon® E5-2650 v3 CPU and 256 GBytes RAM, running on a Linux OS. Due to the randomness, we ran our algorithm 10 times independently with different random seeds (CPU timestamps) for each instance.

We set the different cut-off times T_{cut} (refer to Algorithm 1 line 3 and Algorithm 3 line 4) for instances of different scales, as described below: T_{cut} is set to 2 h for the small scale instances ($n \le 100$); T_{cut} is set to 6 h for the moderate-scale I instances ($101 \le n \le 200$); and T_{cut} is set to 12 h for the moderate-scale II instances ($201 \le n \le 400$). These cut-off time settings are comparable with the state-of-the-art PSO-BA method and other similar works in the literature (e.g., algorithms for circle packing problems). For example, the PSO-BA method set the cut-off time to 2 h for the instances with $n \le 50$; Huang and Yu (2011) obtain the results within the running time of 3,378 s (0.94 h) for $n \le 100$ and 50,982 s (14.16 h) for $101 \le n \le 200$, while solving the PESS problem; Lai et al. (2022) set 2, 8, 12 h for the instances with $n \le 100$, $101 \le n \le 200$, $201 \le n \le 320$, respectively, while solving the packing equal circles in a circle problem.

The default settings for the remaining parameters are described as follows. The maximum iteration step of the heuristic SED is set to $S_{iter} = 700$ (as used in Algorithm 3); the controlling coefficient of function J is set to $c = 7$ (as used in Section 4.3); and the perturbing parameter of the uniform distribution U is set to $\theta = 0.8$ (as used in Section 4.3). The tuning analysis and parameter study are presented in Section 5.4.

5.2. Comparison with state-of-the-art method

The optimal solutions are trivial for $1 \le n \le 4$, and the optimality can be easily proven by mathematical analysis of which the optimal radii are $R_1^* = 1$, $R_2^* = 2$, $R_3^* = 2/\sqrt{3} + 1 \approx 2.1547$, and $R_4^* = \sqrt{6}/2 + 1 \approx 2.2247$. Therefore, we performed our algorithm beginning from $n = 5$. The comparison results of the best-known records reported from Packomania, the state-of-the-art PSO-BA method, and our proposed SED algorithm are shown in Table 1.

In Table 1, the column of n corresponds to the number of packing items in the instance, followed by the best-known record R^* reported on the Packomania website (Specht, 2023). PSO-BA corresponds to the results reported in the work of the state-of-the-art method (Hifi and Yousef, 2018b): R_{best} and R_{avg} indicate the best and average results obtained by PSO-BA over 10 independent runs, and $Time$ (s) indicates the corresponding runtime of R_{best}. SED corresponds to the results of our algorithm: R_{best} and R_{avg} indicate the best and average results obtained by our algorithm over 10 independent runs, and $Time$ (s) indicates the average runtime for SED to obtain its final result. The last two columns of Δ_{best} and Δ_{avg} correspond to the difference in scientific notation between the best and average results of SED and PSO-BA (i.e., $\Delta_{best} = R_{best}^{\mathrm{SED}} - R_{best}^{\mathrm{PSO-BA}}$ and $\Delta_{avg} = R_{avg}^{\mathrm{SED}} - R_{avg}^{\mathrm{PSO-BA}}$), where a negative value indicates that SED yields a better result than PSO-BA. At the bottom of the table, the rows of "#Better", "#Equal", and "#Worse" show the number of instances for which SED obtains the better, equal, and worse results compared with PSO-BA in terms of R_{best} and R_{avg}, respectively.

Note that PSO-BA was performed on an Intel Core 2 Duo (2.53 GHz and with 4Gb of RAM) environment. Since PSO-BA and SED are performed on different computing platforms, the runtime information is only provided for indicative purposes.

From Table 1, we have the following observations:

- SED yields 14 better, 30 equal, and 2 worse results than PSO-BA in terms of R_{best}. The 46 best results obtained by SED reach the best-known records from Packomania with a much shorter computation time. It demonstrates that SED has a strong solving ability on small-scale instances of PESS and significantly outperforms the state-of-the-art PSO-BA method.
- SED yields 30 better, 14 equal, and 2 worse results than PSO-BA in terms of R_{avg}. The 46 average results obtained by SED also reach the best-known records, which means SED has a very high success

Table 1
Computational results and comparison of the best-known records R^* from Packomania, the state-of-the-art PSO-BA method, and our algorithm for the small-scale instances ($3 \le n \le 50$). The best values are presented in bold among the compared results.

n	R^*	PSO-BA			SED (this work)			Δ_{best}	Δ_{avg}
		R_{best}	R_{avg}	$Time$ (s)	R_{best}	R_{avg}	$Time$ (s)		
5	**2.4142135624**	**2.4142135624**	**2.4142135624**	473.91	**2.4142135624**	**2.4142135624**	0.00	0	0
6	**2.4142135624**	**2.4142135624**	**2.4142135624**	66.88	**2.4142135624**	**2.4142135624**	0.00	0	0
7	**2.5912538723**	**2.5912538723**	**2.5912538723**	827.55	**2.5912538723**	**2.5912538723**	0.00	0	0
8	**2.6453287760**	**2.6453287760**	**2.6453287760**	310.62	**2.6453287760**	**2.6453287760**	1.00	0	0
9	**2.7320508076**	**2.7320508076**	**2.7320508076**	206.78	**2.7320508076**	**2.7320508076**	1.00	0	0
10	**2.8324645611**	**2.8324645611**	**2.8324645611**	1290.20	**2.8324645611**	**2.8324645611**	23.30	0	0
11	**2.9021130326**	**2.9021130326**	**2.9021130326**	153.82	**2.9021130326**	**2.9021130326**	1.00	0	0
12	**2.9021130326**	**2.9021130326**	2.9314798912	1179.29	**2.9021130326**	**2.9021130326**	2.00	0	−2.94E−02
13	**3.0000000000**	**3.0000000000**	**3.0000000000**	488.90	**3.0000000000**	**3.0000000000**	4.00	0	0
14	**3.0911454449**	**3.0911454449**	**3.0911454449**	458.23	**3.0911454449**	**3.0911454449**	8.80	0	0
15	**3.1416426249**	**3.1416426249**	**3.1416426249**	6282.91	**3.1416426249**	**3.1416426249**	10.00	0	0
16	**3.2156830320**	**3.2156830320**	3.2157562243	1124.03	**3.2156830320**	**3.2156830320**	5.40	0	−7.32E−05
17	**3.2712455117**	**3.2712455117**	**3.2712455117**	92.19	**3.2712455117**	**3.2712455117**	7.00	0	0
18	**3.3189887817**	**3.3189887817**	**3.3189887817**	707.66	**3.3189887817**	**3.3189887817**	5.00	0	0
19	**3.3860159733**	**3.3860159733**	3.3860161307	6738.12	**3.3860159733**	**3.3860159733**	5.00	0	−1.57E−07
20	**3.4735389622**	**3.4735389622**	3.4748202071	7000.69	**3.4735389622**	**3.4735389622**	164.40	0	−1.28E−03
21	**3.4863514104**	**3.4863514104**	**3.4863514104**	2732.54	**3.4863514104**	**3.4863514104**	6.00	0	0
22	**3.5798331912**	**3.5798331912**	3.5798511534	2500.96	**3.5798331912**	**3.5798331912**	640.40	0	−1.80E−05
23	**3.6275164365**	**3.6275164365**	3.6284425467	5236.08	**3.6275164365**	**3.6275164365**	29.10	0	−9.26E−04
24	**3.6853949355**	**3.6853949355**	3.6861597918	6538.98	**3.6853949355**	**3.6853949355**	205.50	0	−7.65E−04
25	**3.6874267475**	**3.6874267475**	**3.6874267475**	2661.64	**3.6874267475**	**3.6874267475**	185.80	0	0
26	**3.7474057765**	**3.7474057765**	3.7474079315	1805.77	**3.7474057765**	**3.7474057765**	9.00	0	−2.16E−06
27	**3.8134159569**	**3.8134159569**	3.8154473167	4003.44	**3.8134159569**	**3.8134159569**	266.80	0	−2.03E−03
28	**3.8416402781**	**3.8416402781**	3.8427270479	6877.24	**3.8416402781**	**3.8416402781**	580.20	0	−1.09E−03
29	**3.8770891032**	**3.8770891032**	3.8773011833	6802.59	**3.8770891032**	**3.8770891032**	561.70	0	−2.12E−04
30	**3.9164916616**	3.9164916887	3.9182908180	6661.05	**3.9164916616**	**3.9164916616**	349.70	−2.71E−08	−1.80E−03
31	**3.9507544849**	**3.9507544849**	3.9508604111	6794.01	**3.9507544849**	**3.9507544849**	1639.20	0	−1.06E−04
32	**3.9874403893**	**3.9874403893**	3.9885139480	6783.05	**3.9874403893**	**3.9874403893**	309.90	0	−1.07E−03
33	**4.0199009160**	4.0199009254	4.0200538386	6897.21	**4.0199009160**	**4.0199009160**	633.20	−9.40E−09	−1.53E−04
34	**4.0477199712**	**4.0477199712**	4.0492757675	5846.87	**4.0477199712**	**4.0477199712**	436.90	0	−1.56E−03
35	**4.0844057408**	4.0844077658	4.0854592377	6852.70	**4.0844057408**	**4.0844057408**	2410.10	−2.02E−06	−1.05E−03
36	**4.1129893297**	**4.1129893297**	4.1284894708	5000.74	**4.1129893297**	**4.1129893297**	322.30	0	−1.55E−02
37	**4.1547812520**	4.1548030417	4.1564599544	7045.52	**4.1547812520**	**4.1547812520**	4287.00	−2.18E−05	−1.68E−03
38	**4.1576692600**	4.1576724213	4.1578290898	6661.83	**4.1576692600**	**4.1576692600**	7200.00	−3.16E−06	−1.60E−04
39	**4.2239497563**	**4.2239497563**	4.2271089635	6833.07	**4.2239497563**	**4.2239497563**	631.70	0	−3.16E−03
40	**4.2553329537**	4.2553331430	4.2568592733	6588.66	**4.2553329537**	**4.2553329537**	569.40	−1.89E−07	−1.53E−03
41	4.2963450048	**4.2553329537**	4.2844863118	7198.97	4.2963450048	4.2963450048	1197.90	4.10E−02	1.19E−02
42	4.3081420430	**4.2553329537**	4.2893547713	5984.61	4.3081420430	4.3081420430	33.00	5.28E−02	1.88E−02
43	**4.3528798324**	4.3530264887	4.3576909767	5153.70	**4.3528798324**	**4.3528798324**	2055.90	−1.47E−04	−4.81E−03
44	**4.3828308379**	4.3828454090	4.3845814567	6339.39	**4.3828308379**	**4.3828308379**	6145.30	−1.46E−05	−1.75E−03
45	**4.4070031477**	4.4070031605	4.4118673635	6762.92	**4.4070031477**	**4.4070031477**	1397.60	−1.28E−08	−4.86E−03
46	**4.4411244747**	4.4417854293	4.4449129285	6146.29	**4.4411244747**	**4.4411244747**	1232.30	−6.61E−04	−3.79E−03
47	**4.4741318035**	4.4744654770	4.4769235843	5523.97	**4.4741318035**	**4.4741318035**	1531.70	−3.34E−04	−2.79E−03
48	**4.4962827447**	4.4963283111	4.4975725203	5194.01	**4.4962827447**	**4.4962827447**	6323.20	−4.56E−05	−1.29E−03
49	**4.5191984746**	4.5192891866	4.5217695981	7102.21	**4.5191984746**	**4.5191984746**	2341.90	−9.07E−05	−2.57E−03
50	**4.5504543407**	4.5509544053	4.5517945731	6346.64	**4.5504543407**	**4.5504543407**	974.70	−5.00E−04	−1.34E−03
#Better					14	30			
#Equal					30	14			
#Worse					2	2			

rate in obtaining the best result on small-scale instances. It also demonstrates that SED significantly outperforms PSO-BA, and SED is an efficient and powerful heuristic algorithm for solving PESS that can stably obtain most best solutions on small-scale PESS instances.

5.3. Comparison with best-known records

We further evaluate our proposed SED algorithm's performance on small and moderate scale instances in the range of $5 \le n \le 400$ by making a comparison with the best-known record R^* reported on the Packomania website (Specht, 2023). Tables 2 and 3 and Fig. 3 summarize the computational results and comparison of our algorithm and the results of Packomania on the PESS problem. Meanwhile, we also provide the detailed computational results and comparison in Appendix A.1, including the average and worst results obtained by SED over 10 independent runs, the difference between the best-known records and our algorithm's results, and the average runtime of 10 runs of SED, etc.

Table 2 shows the 274 improved results[1] obtained by SED compared with the best-known records from Packomania. The column of n corresponds to the number of packing items in the instance, the column of R^* shows the best-known records on the Packomania website, and the column of R_{best} shows the best results obtained by SED over 10 independent runs in this work. The improved results appear in bold in the table.

Table 3 makes a summary of our algorithm's results on the PESS problem in the range of $5 \le n \le 400$. It shows that our algorithm yields the number of improved, equal, and worse results compared with the best-known records in eight different scale intervals, and the total

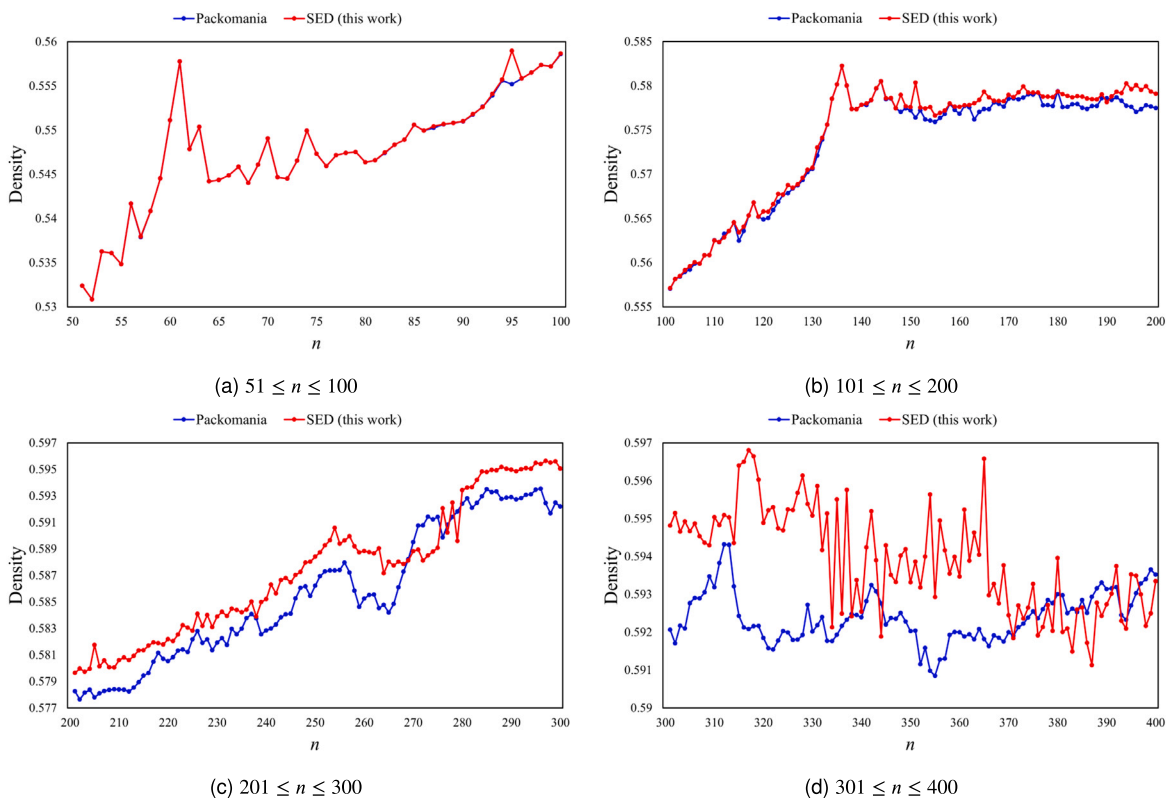
[1] Note that our improved results have been published on the Packomania website.

Table 2
Improved results for 274 instances in the range of $5 \le n \le 400$ in terms of R_{best} compared with the best-known records R^* from Packomania, and the improved results appear in bold.

n	R^*	R_{best}	n	R^*	R_{best}	n	R^*	R_{best}	n	R^*	R_{best}
57	4.7322376049	**4.7319976099**	162	6.5459496371	**6.5450672093**	233	7.3660814854	**7.3597164787**	311	8.0606039399	**8.0560797869**
73	5.1117410063	**5.1117310540**	163	6.5645506055	**6.5577283543**	234	7.3783260883	**7.3706429647**	312	8.0669842552	**8.0634879963**
76	5.1827544049	**5.1827522689**	164	6.5747462297	**6.5695683516**	235	7.3870071716	**7.3818305285**	313	8.0756527378	**8.0723776565**
77	5.2014502549	**5.2014494794**	165	6.5868295847	**6.5794341232**	236	7.3941962819	**7.3915014587**	314	8.0892357495	**8.0840444646**
80	5.2707282625	**5.2707228597**	166	6.6002228950	**6.5951677305**	237	7.4033253417	**7.3993046206**	315	8.1013721248	**8.0833362248**
81	5.2918921098	**5.2918269759**	167	6.6104638217	**6.6098597108**	238	7.4151037322	**7.4144807701**	316	8.1113304407	**8.0914438974**
82	5.3108551826	**5.3107500414**	168	6.6242823896	**6.6230917029**	239	7.4305286960	**7.4202453485**	317	8.1200535298	**8.0985648551**
84	5.3488647209	**5.3488512301**	169	6.6386257645	**6.6363389670**	240	7.4396535786	**7.4295984609**	318	8.1282408602	**8.1078206147**
87	5.4073256708	**5.4067811877**	170	6.6482796492	**6.6466042999**	241	7.4493181639	**7.4352855259**	319	8.1367023254	**8.1191101842**
88	5.4266705368	**5.4266460069**	171	6.6610392083	**6.6606281796**	242	7.4583263208	**7.4483780897**	320	8.1466752518	**8.1327700764**
90	5.4663957267	**5.4663528574**	172	6.6745006321	**6.6714601988**	243	7.4662779719	**7.4543237759**	321	8.1563823395	**8.1397394926**
91	5.4840116687	**5.4838608716**	173	6.6866891682	**6.6818155183**	244	7.4755562089	**7.4638668575**	322	8.1650140248	**8.1477840758**
93	5.5167123871	**5.5161208996**	174	6.6981105869	**6.6973820953**	245	7.4855419380	**7.4754936428**	323	8.1723756268	**8.1587592542**
94	5.5307641333	**5.5304774705**	175	6.7110188626	**6.7101346800**	246	7.4908312394	**7.4833047976**	324	8.1796327858	**8.1674085081**
95	5.5516834671	**5.5391547512**	176	6.7228800450	**6.7228629814**	247	7.4975449341	**7.4923815998**	325	8.1882217082	**8.1733003234**
96	5.5689451565	**5.5689435455**	177	6.7412137384	**6.7374499882**	248	7.5072043746	**7.4995291317**	326	8.1975179838	**8.1817407277**
98	5.6022822809	**5.6022558001**	178	6.7537260778	**6.7500785679**	249	7.5203328104	**7.5092769099**	327	8.2058236905	**8.1880247293**
99	5.6218126153	**5.6217725678**	179	6.7668261653	**6.7629223995**	250	7.5270535066	**7.5177181629**	328	8.2136443647	**8.1942670899**
100	5.6359808164	**5.6357325219**	181	6.7924484211	**6.7866497775**	251	7.5340290155	**7.5263662601**	329	8.2182864526	**8.2060215573**
101	5.6599579629	**5.6597794928**	182	6.8047492229	**6.7999756316**	252	7.5424648986	**7.5340412095**	330	8.2298817657	**8.2157405956**
103	5.6923913336	**5.6921248921**	183	6.8160062463	**6.8129617420**	253	7.5521308100	**7.5424247553**	331	8.2373988670	**8.2204413597**
104	5.7089561454	**5.7082798932**	184	6.8283189424	**6.8249182583**	254	7.5621077128	**7.5482890171**	332	8.2447022853	**8.2365077203**
105	5.7263005035	**5.7250734932**	185	6.8424112657	**6.8374332956**	255	7.5718630254	**7.5633379207**	333	8.2558752036	**8.2402967266**
106	5.7420854884	**5.7416014823**	186	6.8552427473	**6.8504870996**	256	7.5793149724	**7.5721483921**	334	8.2641607509	**8.2624688079**
108	5.7747329463	**5.7747315860**	187	6.8661850359	**6.8631014993**	257	7.5923599786	**7.5806305162**	335	8.2716242498	**8.2550643788**
111	5.8226237949	**5.8225938205**	188	6.8783760115	**6.8753773078**	258	7.6081822216	**7.5937072918**	336	8.2787743871	**8.2772630110**
114	5.8668408971	**5.8668377535**	189	6.8869865649	**6.8853029301**	259	7.6232794657	**7.6054822855**	337	8.2862241400	**8.2702833068**
115	5.8911773387	**5.8877955184**	191	6.9119625946	**6.9104619032**	260	7.6303474460	**7.6147896386**	339	8.3019941224	**8.2976655457**
116	5.9043232575	**5.9026298032**	192	6.9229251057	**6.9203258423**	261	7.6389568630	**7.6249382747**	340	8.3104116379	**8.3097262711**
117	5.9151425496	**5.9151356434**	193	6.9364631911	**6.9329148694**	262	7.6485119939	**7.6350999584**	341	8.3165790385	**8.3099338147**
119	5.9491701550	**5.9490619877**	194	6.9506991452	**6.9405887493**	263	7.6627443277	**7.6430467452**	342	8.3227045402	**8.3135903730**
120	5.9668074819	**5.9636279898**	195	6.9631085477	**6.9551013467**	264	7.6713523051	**7.6609426956**	343	8.3315888726	**8.3277561232**
121	5.9827857339	**5.9802333759**	196	6.9772480051	**6.9651197544**	265	7.6835789969	**7.6668328972**	345	8.3518472673	**8.3420522869**
122	5.9960642820	**5.9936598394**	197	6.9878958414	**6.9790672127**	266	7.6903773179	**7.6776988774**	346	8.3591153674	**8.3537759755**
123	6.0090785558	**6.0059664994**	198	6.9979276863	**6.9892007374**	267	7.6944927612	**7.6860055622**	347	8.3672718772	**8.3627048726**
124	6.0225351062	**6.0224402824**	199	7.0101428170	**7.0034867783**	268	7.6988832962	**7.6964397347**	348	8.3745515295	**8.3674634360**
125	6.0380168643	**6.0347337708**	200	7.0226524557	**7.0160475056**	276	7.7633757754	**7.7537790480**	349	8.3836313552	**8.3746367225**
126	6.0521543613	**6.0519428829**	201	7.0311579735	**7.0255544330**	278	7.7753213557	**7.7705949667**	350	8.3928242707	**8.3867647460**
127	6.0668536575	**6.0665232674**	202	7.0452775617	**7.0358154477**	280	7.7895974859	**7.7850390707**	351	8.4007527076	**8.3921705143**
128	6.0806366813	**6.0797285101**	203	7.0547724371	**7.0484216078**	281	7.7970295470	**7.7935255540**	352	8.4129454678	**8.4033387897**
129	6.0932666869	**6.0922757381**	204	7.0654812476	**7.0590273102**	282	7.8094475872	**7.8025455856**	353	8.4188352568	**8.4074380625**
130	6.1076531690	**6.1072029065**	205	7.0793921434	**7.0632644486**	283	7.8169572685	**7.8094014911**	354	8.4296585287	**8.4076401001**
131	6.1178856994	**6.1146523017**	206	7.0896011259	**7.0812834485**	284	7.8239824233	**7.8157677410**	355	8.4382375759	**8.4283461812**
132	6.1269393558	**6.1262395723**	207	7.1003436731	**7.0909103971**	285	7.8308005585	**7.8251346857**	356	8.4440909450	**8.4267054911**
133	6.1363940823	**6.1363253673**	208	7.1114371819	**7.1043854402**	286	7.8409629540	**7.8335735779**	357	8.4518644222	**8.4382769178**
134	6.1412839494	**6.1412827373**	209	7.1225774947	**7.1158587077**	287	7.8498687648	**7.8428652701**	358	8.4567949768	**8.4490841743**
136	6.1584969194	**6.1584884950**	210	7.1340172788	**7.1249463415**	288	7.8614608232	**7.8508039705**	359	8.4642649260	**8.4548490764**
138	6.2059663809	**6.2059641354**	211	7.1453393266	**7.1353927996**	289	7.8700993136	**7.8605264864**	360	8.4721516869	**8.4651558516**
139	6.2210255874	**6.2209771473**	212	7.1571882237	**7.1491071717**	290	7.8790360850	**7.8698725201**	361	8.4805536014	**8.4645834364**
140	6.2341312659	**6.2341210439**	213	7.1672317136	**7.1573496074**	291	7.8887911114	**7.8793971860**	362	8.4880679606	**8.4788009361**
141	6.2490415957	**6.2483731721**	214	7.1767228307	**7.1668129415**	292	7.8973927600	**7.8877228593**	363	8.4965064991	**8.4830627684**
143	6.2716966903	**6.2715956530**	215	7.1858107754	**7.1779075067**	293	7.9052966809	**7.8963644352**	364	8.5030156544	**8.4936601180**
144	6.2832371987	**6.2832364663**	216	7.1962064445	**7.1876566450**	294	7.9140909168	**7.9055450950**	365	8.5120741140	**8.4893661721**
145	6.3051401291	**6.3048057891**	217	7.2037038757	**7.1977035204**	295	7.9215067329	**7.9125134312**	366	8.5207116927	**8.5142845330**
146	6.3192938478	**6.3191616112**	218	7.2119969745	**7.2089282580**	296	7.9301910270	**7.9218499738**	367	8.5271184405	**8.5206107101**
148	6.3535817257	**6.3465377175**	219	7.2248960523	**7.2203773197**	297	7.9439217715	**7.9296931750**	368	8.5351429761	**8.5308028271**
149	6.3663869913	**6.3654709825**	220	7.2365800516	**7.2296491612**	298	7.9562684663	**7.9392349019**	369	8.5433733193	**8.5337041371**
150	6.3814711453	**6.3800749408**	221	7.2462659092	**7.2412540521**	299	7.9615077588	**7.9476166544**	370	8.5498955957	**8.5477264184**
151	6.3985999459	**6.3840816057**	222	7.2551034048	**7.2500633764**	300	7.9716783431	**7.9589157181**	372	8.5646395490	**8.5618715156**
152	6.4096321832	**6.4084860226**	223	7.2656309049	**7.2580609579**	301	7.9811692972	**7.9688310111**	373	8.5718304380	**8.5712024323**
153	6.4275041177	**6.4229011359**	224	7.2773063486	**7.2696697784**	302	7.9916010741	**7.9761636886**	374	8.5786934269	**8.5774728308**
154	6.4419628019	**6.4363632879**	225	7.2841046441	**7.2813658072**	303	7.9983093317	**7.9871618864**	375	8.5855784311	**8.5820685572**
155	6.4564928744	**6.4537179487**	226	7.2923021211	**7.2868642085**	304	8.0074253659	**7.9947232300**	380	8.6213515024	**8.6167407193**
156	6.4687030989	**6.4664665398**	227	7.3067387277	**7.3013688606**	305	8.0131815877	**8.0046510979**	384	8.6537986688	**8.6534665127**
157	6.4806879723	**6.4793187351**	228	7.3163529791	**7.3086523632**	306	8.0213078802	**8.0124605464**	392	8.7102535414	**8.7075529636**
158	6.4901644711	**6.4899861268**	229	7.3304940328	**7.3230948777**	307	8.0301086572	**8.0226883734**	395	8.7348094475	**8.7307593517**
159	6.5063519438	**6.5050243930**	230	7.3387673249	**7.3304753817**	308	8.0380891717	**8.0321834308**	396	8.7405814827	**8.7383102973**
160	6.5216146623	**6.5187621450**	231	7.3478959531	**7.3395741975**	309	8.0448618089	**8.0411618615**			
161	6.5318675905	**6.5315929690**	232	7.3606440021	**7.3514522524**	310	8.0548522945	**8.0464675478**			

Table 3
Summary of the computational results of our SED algorithm in the range of $5 \le n \le 400$.

	n								Total	Ratio
	[5, 50]	[51, 100]	[101, 150]	[151, 200]	[201, 250]	[251, 300]	[301, 350]	[351, 400]		
#Improved	0	19	39	48	50	41	48	29	274	69.19%
#Equal	46	30	8	0	0	0	0	0	84	21.21%
#Worse	0	1	3	2	0	9	2	21	38	9.60%



(a) $51 \le n \le 100$ (b) $101 \le n \le 200$ (c) $201 \le n \le 300$ (d) $301 \le n \le 400$

Fig. 3. Packing density plots for the best-known records from Packomania and the best results of SED (this work).

number and the ratio of improved, equal, and worse results are counted in the last two columns.

For an intuitive comparing purpose, we further provide a packing density plot of the results of Packomania and SED, which is presented in Fig. 3, and the density of a configuration of the PESS problem is formulated in Eq. (6). The four subplots show the comparison on the four different scales [51, 100], [101, 200], [201, 300], and [301, 400], respectively, where the interval [5, 50] is omitted because the results obtained by SED are the same as the results reported from Packomania. In each subplot, the X-axis indicates the number of packing items, and the Y-axis indicates the packing density of a feasible configuration. Note that a higher density indicates a better configuration.

From Tables 2 and 3 and Fig. 3, we have the following observations.

(1) On small-scale instances. SED yields 19 improved results compared with the best-known records, matches the best-known records for 76 instances and misses the best-known records for only one instance ($n = 92$). It demonstrates that our algorithm has a very strong solving ability and can always obtain a dense configuration on small-scale PESS instances.

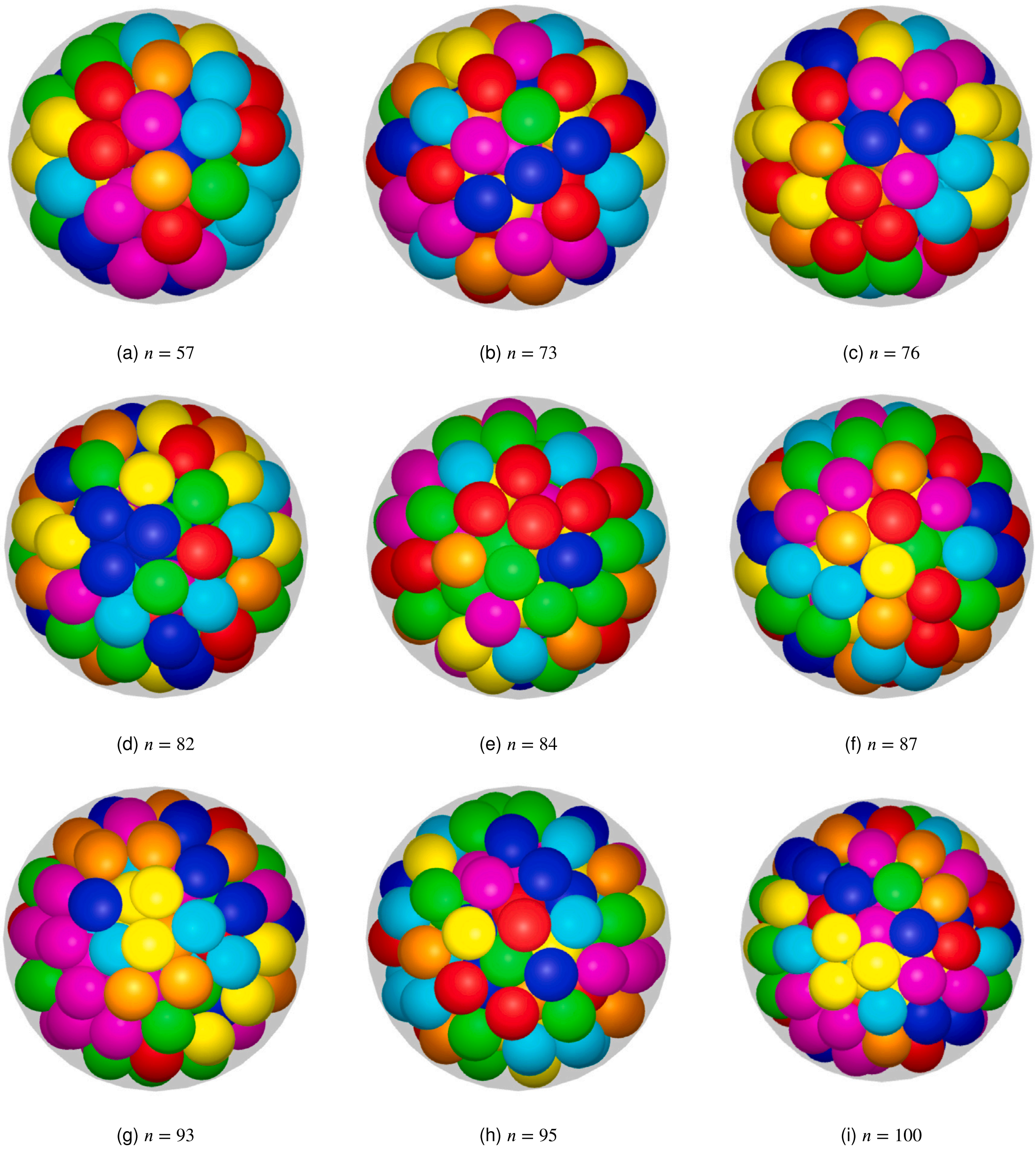
(2) On moderate-scale I instances. SED yields 87 improved, 8 equal, and 5 worse results compared with the best-known records on this scale. The packing density plot of the interval [101, 200] shows that SED gains many improvements, even if some of them are very slight, implying that the configuration on the Packomania website is dense enough, but SED also can obtain a denser configuration for $101 \le n \le 200$. It demonstrates that our algorithm exhibits excellent performance on moderate-scale PESS instances.

(3) On moderate-scale II instances. SED yields 168 improved, no equal, and 32 worse results compared with the best-known records on this scale. The packing density plot of the interval [201, 400] shows that SED gains a lot of significant improvements in these instances, and the worse results are mainly located in the interval [371, 400], which might be caused by the number of packing items being too large in this interval and the cut-off time of 12 h cannot support SED to yield an improved result. These experimental results also demonstrate that our algorithm has an excellent performance on moderate-scale PESS instances.

As discussed above, SED is a powerful and efficient heuristic, and our algorithm has an excellent performance for solving the PESS problem. In addition, Figs. 4–6 provide some graphical configurations of the improved solutions obtained by our algorithm selected from each scale.

5.4. Parameter study

Three parameters need to be tuned in the algorithm, including the maximum iteration step S_{iter} of the SED heuristic (see in Algorithm



(a) $n = 57$ (b) $n = 73$ (c) $n = 76$

(d) $n = 82$ (e) $n = 84$ (f) $n = 87$

(g) $n = 93$ (h) $n = 95$ (i) $n = 100$

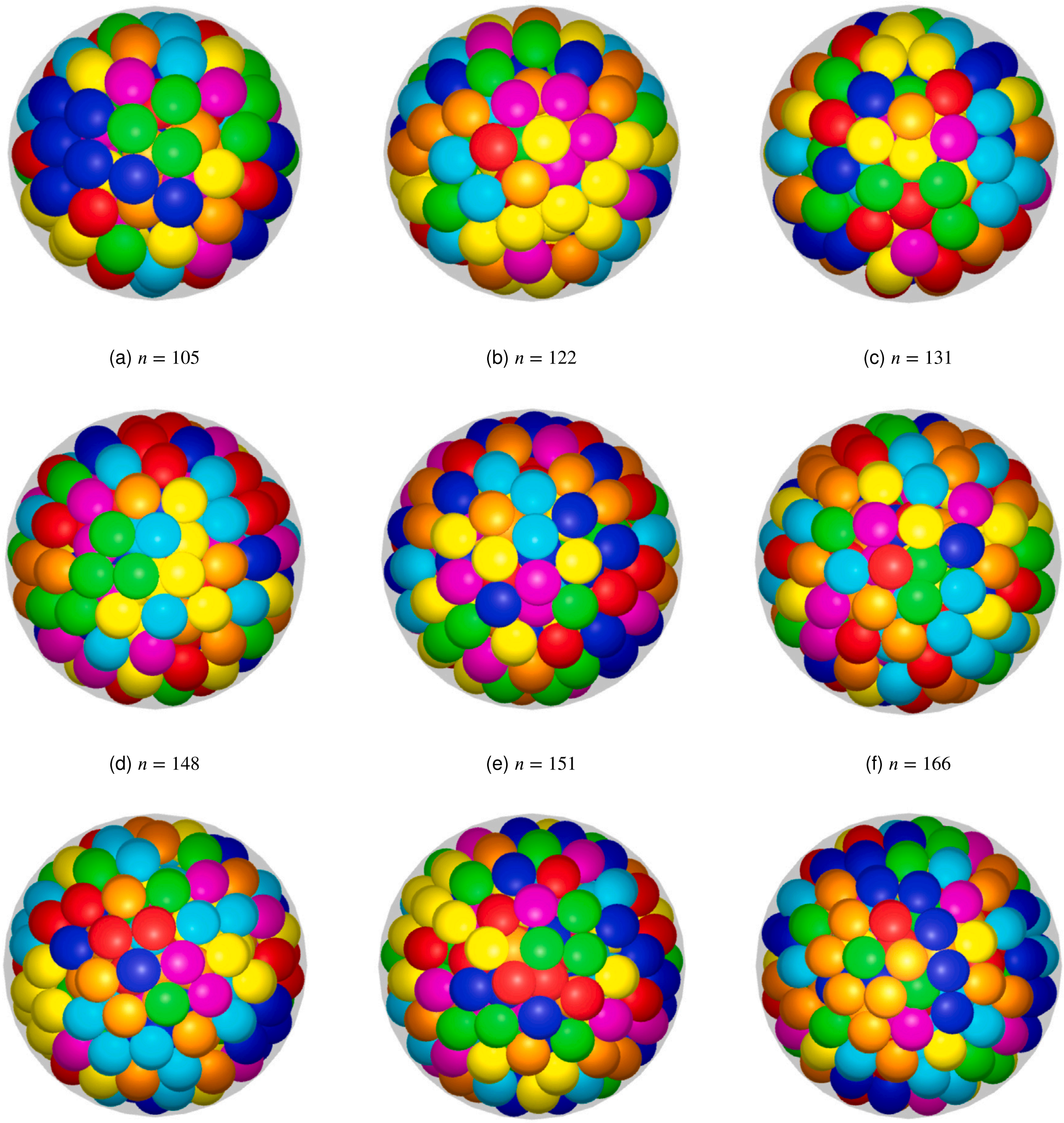
Fig. 4. Improved solutions found by our algorithm sampled from the small-scale instances ($5 \le n \le 100$).

3), the controlling coefficient c of the function J (see in Section 4.3), and the perturbing parameter θ of the uniform distribution U (see in Section 4.3). We randomly select 2, 4, and 8 instances from the small, moderate I, and moderate II scales, respectively. We tune the three parameters on the 14 selected test instances. The tuning experiments and parameter study are presented as follows.

On the maximum iteration step of SED. We perform our algorithm with different maximum iteration step settings of the SED heuristic for $S_{iter} = 500, 600, \dots, 1000$ on the 14 selected instances, and the other parameters are set as default ($c = 7$ and $\theta = 0.8$). Experimental results are summarized in Table 4, the column of n shows the number of packing spheres of the instance, columns 2-7 show the average results R_{avg} of the algorithm over 10 independent runs for each parameter setting S_{iter} on the 14 selected instances. The last row "Average" shows the average value of the 14 average results of each column.

From Table 4, one can observe that the setting $S_{iter} = 500$ gains the most of best results in terms of R_{avg} among the 6 tested parameter settings for 6 out of the 14 selected instances, followed by the settings $S_{iter} = 700$, 600, 800, 900 and 1000 gain the number of best results for 5, 3, 2, 2 and 1 out of the 14 selected instances. However, the setting $S_{iter} = 700$ gains the best average value among the 6 tested settings. As the setting $S_{iter} = 700$ is only one less than the setting $S_{iter} = 500$ of the best result, but it gains the best average value, so we choose $S_{iter} = 700$ as the default setting.

On the controlling coefficient. The parameter c is a coefficient of the function J (Eq. (7)) to control the size of perturbing candidate set C. We perform our algorithm with different controlling coefficient settings for $c = 5, 6, \dots, 9$ on the 14 selected instances, and the other parameters are set as default ($S_{iter} = 700$ and $\theta = 0.8$). Experimental results are summarized in Table 5, the column of n shows the number of packing spheres of the instance, columns 2-6 show the average results R_{avg} of the algorithm over 10 independent runs for each parameter setting c on the 14 selected instances. The last row "Average" shows the average value of the 14 average results of each column.



(a) $n = 105$ (b) $n = 122$ (c) $n = 131$

(d) $n = 148$ (e) $n = 151$ (f) $n = 166$

(g) $n = 173$ (h) $n = 183$ (i) $n = 196$

Fig. 5. Improved solutions found by our algorithm sampled from the moderate-scale I instances ($101 \leq n \leq 200$).

From Table 5, we have the following observations. The setting of $c = 6$ and 7 gains the most of best results in terms of R_{avg} among the 5 tested parameter settings for 6 out of the 14 selected instances, followed by the settings $c = 9$, 5 and 8 gain the number of 3, 2 and 1 out of the 14 selected instances. And the setting $c = 7$ gains the best average value among the 5 tested settings. Therefore, we choose $c = 7$ as the default setting.

On the perturbing parameter. The perturbing parameter θ of the uniform distribution U controls the random shifting value of packing spheres' coordinates in the perturbing operator. We perform our algorithm with different perturbing parameter settings for $\theta = 0.6, 0.7, \ldots, 1.0$ on the 14 selected instances, and the other parameters are set as default ($S_{iter} = 700$ and $c = 7$). Experimental results are summarized in Table 6, the column of n shows the number of packing spheres of the instance, columns 2-6 show the average results R_{avg} of the algorithm over 10 independent runs for each parameter setting θ on the 14 selected instances, and the last row "Average" shows the average value of the 14 average results of each column.

Table 6 shows the algorithm with the setting $\theta = 0.8$ gains the best performance in terms of R_{avg} for 13 out of the 14 selected instances, which is obviously better than the other 4 tested parameter settings. It also gains the best average value among the 5 tested parameter settings. Therefore, we choose $\theta = 0.8$ as the default setting.

5.5. Limitation and performance on cut-off times

In Table 3, we observe that most of the worse results compared with Packomania are located in the interval of $[351, 400]$. We are concerned about two reasons for this performance: (1) Limited by its solving ability, it is hard for SED to yield better results on these moderate-scale instances; (2) The PESS problem is computationally very challenging, and SED requests a sufficient running time to yield an improved result on such number of packing items of the instance.

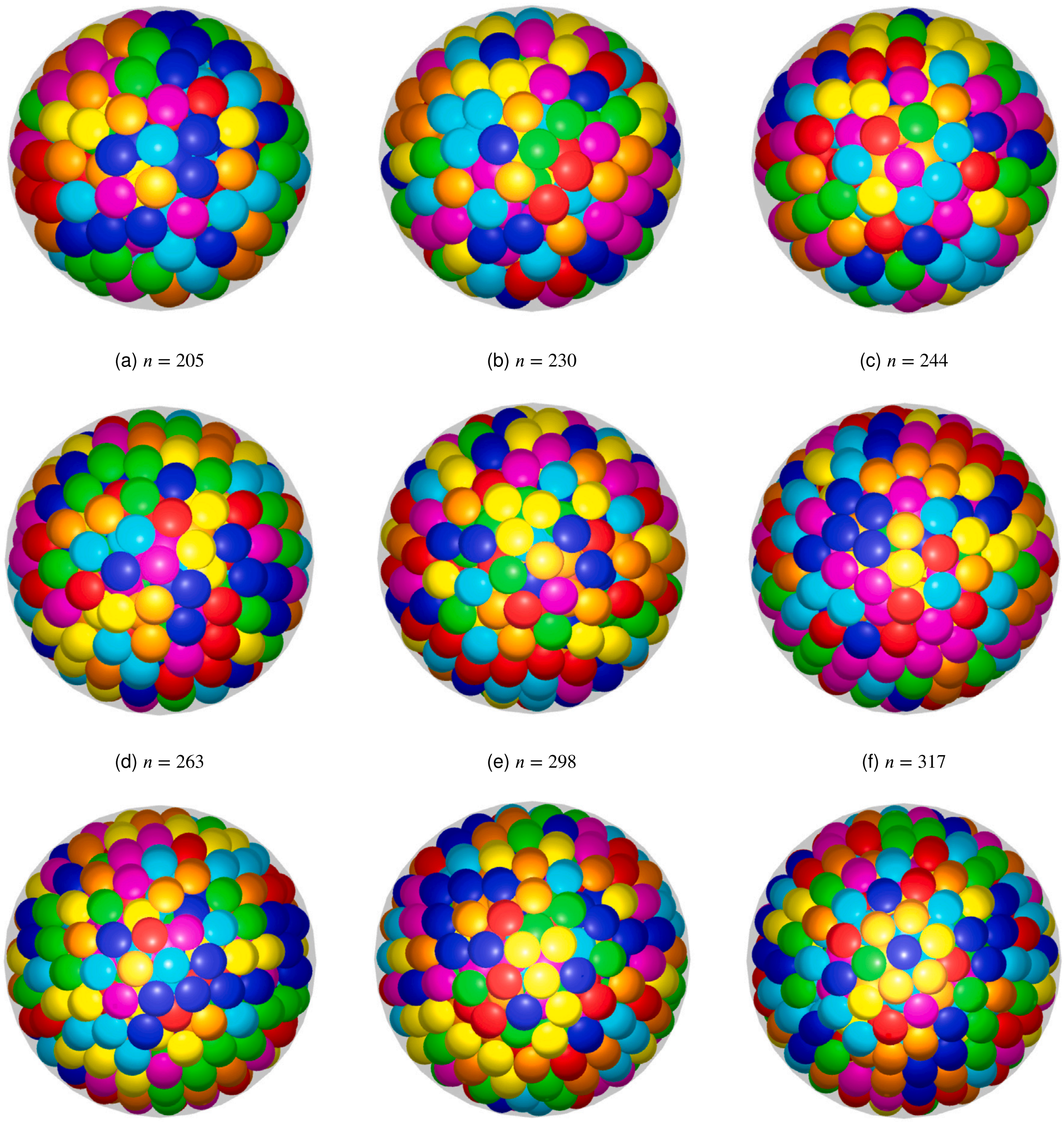

(a) $n = 205$ (b) $n = 230$ (c) $n = 244$
(d) $n = 263$ (e) $n = 298$ (f) $n = 317$
(g) $n = 355$ (h) $n = 361$ (i) $n = 395$

Fig. 6. Improved solutions found by our algorithm sampled from the moderate-scale II instances ($201 \le n \le 400$).

Therefore, we conducted an additional experiment to analyze the limitations of our algorithm. We reperformed our algorithm on the 50 moderate-scale instances with $351 \le n \le 400$ and set the cut-off time to 24 h. The algorithm reperformed 10 independent runs on each instance, which is similar to the experiments in Sections 5.2 and 5.3, and the other parameters are set as default. To further analyze the performance of our algorithm, we recorded the best results on the different time points of 2, 6, and 12 h during each run, which can be regarded as different results on different cut-off times. The results of the additional experiment and the comparison are summarized in Table 7 and Fig. 7. The detailed computational results and comparison are provided in Appendix A.2.

Table 7 shows a comparison of the best-known records on the Packomania website and the best results obtained by SED with the four different cut-off times on the 50 moderate-scale instances ($351 \le n \le 400$). The column of n corresponds to the number of packing items in the instance. The column of R^* shows the best-known records from Packomania. T_{cut} shows the results obtained by SED in which the intermediate results recorded on the three different time points are shown in 2 h, 6 h, and 12 h, respectively, and the final results are shown in 24 h. The column of R_{best} shows the best results over 10 independent runs for each cut-off time and the column of Δ_{best} shows the difference in scientific notation between the best result and the best-known record, i.e., $\Delta_{best} = R_{best} - R^*$ where a negative value indicates that SED yields an improved result compared with the best-known record. The improved results appear in bold. At the bottom of the table, the three rows of "#Better", "#Equal", and "#Worse" show the number of instances for which SED obtains the better, equal, and worse results compared with the best-known records.

For an intuitive comparing purpose, the packing density plot of the best-known records and the results of the four different cut-off times of our algorithm on the 50 moderate-scale instances ($351 \le n \le 400$) are presented in Fig. 7, and the density of a configuration of the PESS problem is formulated in Eq. (6). In the plot, the X-axis indicates the

Table 4
Computational results on the average result R_{avg} with the comparison of the maximum iteration step S_{iter} of the SED heuristic for the 14 selected instances where the best results obtained among the tested parameter values are presented in bold.

n \ S_{iter}	500	600	700	800	900	1000
60	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**
85	5.3685586845	5.3675428274	**5.3670410532**	5.3670476557	5.3685604097	5.3695772339
125	**6.0353740583**	6.0365514182	6.0356903597	6.0356615862	6.0365558169	6.0356733701
140	**6.2341210801**	6.2341254599	6.2342623742	6.2351625245	6.2353866960	6.2353877254
177	**6.7386089529**	6.7388866702	6.7390210504	6.7391547201	6.7393323039	6.7389221430
199	**7.0059307131**	7.0079084917	7.0065604316	7.0076963008	7.0072536007	7.0071925355
205	7.0719017567	7.0714566652	**7.0702898675**	7.0715287949	7.0734399411	7.0725053780
212	7.1499807252	7.1501900954	**7.1490794335**	7.1492131460	7.1499729965	7.1495631426
239	7.4278085363	**7.4241903991**	7.4256091484	7.4258169196	7.4278516343	7.4272433070
257	7.5829603512	7.5830260777	**7.5823156930**	7.5828835977	7.5832012629	7.5844114527
285	7.8300728675	7.8356930128	7.8311060285	7.8360062185	**7.8262750705**	7.8340063708
316	8.1005066257	8.1006066623	8.0974417541	**8.0974261420**	8.1008944597	8.1045266682
364	8.5053879884	**8.5010834702**	8.5015802313	8.5026067074	8.5048700031	8.5047659284
398	**8.7635298355**	8.7641287155	8.7635387623	8.7636618905	8.7645112445	8.7658463846
Average	7.0421196976	7.0421659683	**7.0413192699**	7.0420571282	7.0423599307	7.0431825165

Table 5
Computational results on the average result R_{avg} with the comparison of the controlling coefficient c for the 14 selected instances where the best results obtained among the tested parameter values are presented in bold.

n \ c	5	6	7	8	9
60	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**
85	5.3659779921	**5.3653440298**	5.3670410532	5.3685624347	5.3680578744
125	6.0359878384	**6.0352374204**	6.0356903597	6.0358054004	6.0358484826
140	**6.2341290526**	6.2341662886	6.2342623742	6.2353887857	6.2353827882
177	6.7391359616	**6.7388725013**	6.7390210504	6.7391934704	6.7398805469
199	7.0070497619	**7.0064735311**	7.0065604316	7.0080724710	7.0069524091
205	7.0725672377	7.0736081776	**7.0702898675**	7.0725855196	7.0706347878
212	7.1512715786	7.1502877988	**7.1490794335**	7.1496716173	7.1506760487
239	7.4292899636	7.4282991831	**7.4256091484**	7.4272586794	7.4262519789
257	7.5841140578	**7.5820822034**	7.5823156930	7.5823303822	7.5861163752
285	7.8313928737	7.8345924449	**7.8311060285**	7.8354999162	7.8435577572
316	8.1038412345	8.0990948706	**8.0974417541**	8.0974765499	8.1045383268
364	8.5046370926	8.5031591953	8.5015802313	8.5031584842	**8.4994721948**
398	8.7670891784	8.7658106100	8.7635387623	8.7641103181	**8.7630427590**
Average	7.0429583867	7.0422829889	**7.0413192699**	7.0424319728	7.0432389943

Table 6
Computational results on the average result R_{avg} with the comparison of the perturbing parameter θ for the 14 selected instances where the best results obtained among the tested parameter values are presented in bold.

n \ θ	0.6	0.7	0.8	0.9	1.0
60	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**	**4.7749335903**
85	5.3699068636	5.3690764628	**5.3670410532**	5.3670841519	5.3674077272
125	6.0405877816	6.0367733617	**6.0356903597**	6.0373515386	6.0418923290
140	6.2402073454	6.2345660948	**6.2342623742**	6.2347088694	6.2353349643
177	6.7479598125	6.7422245246	**6.7390210504**	6.7424724758	6.7516510345
199	7.0147025938	7.0071138997	**7.0065604316**	7.0198344634	7.0210409221
205	7.0829570570	7.0774460733	**7.0702898675**	7.0868646404	7.0886441012
212	7.1555605981	7.1506092628	**7.1490794335**	7.1633692354	7.1643829465
239	7.4325759003	7.4277873712	**7.4256091484**	7.4412882629	7.4417633709
257	7.6062088590	7.5961504907	**7.5823156930**	7.6128361575	7.6163300946
285	7.8562965133	7.8394809703	**7.8311060285**	7.8655086824	7.8715087731
316	8.1216621740	8.1152536425	**8.0974417541**	8.1343618238	8.1375496725
364	8.5111844242	8.5026266687	**8.5015802313**	8.5216024571	8.5245154455
398	8.7606948783	**8.7551382934**	8.7635387623	8.7737582321	8.7743796038
Average	7.0511027422	7.0449414791	**7.0413192699**	7.0554267558	7.0579524697

number of packing items, and the Y-axis indicates the packing density of a feasible configuration. Note that a higher density indicates a better configuration.

From Table 7 and Fig. 7, we have the following observations. Whether SED can yield an improved solution on moderate-scale PESS instances mainly depends on the running time. For instance, in the tested 50 moderate-scale instances, SED can only yield 13 improved results in 2 h, but as the running time increases, SED can yield 30 improved results in 6 h and 44 improved results in 12 h. Finally, SED can improve all the best-known records on the tested 50 moderate-scale instances in 24 h. Besides, the quality of the solutions obtained in 24 h is significantly better than the solutions obtained in 12 h. The same

Table 7
Computational results and comparison of the best-known records R^* from Packomania and our algorithm with several cut-off times for the moderate-scale instances ($351 \le n \le 400$). The better results appear in bold compared with the best-known records.

n	R^*	T_{cut}							
		2 h		6 h		12 h		24 h	
		R_{best}	Δ_{best}	R_{best}	Δ_{best}	R_{best}	Δ_{best}	R_{best}	Δ_{best}
351	8.4007527076	**8.3996465688**	−1.11E−03	**8.3991945364**	−1.56E−03	**8.3945154809**	−6.24E−03	**8.3940564303**	−6.70E−03
352	8.4129454678	**8.4091364930**	−3.81E−03	**8.4088235272**	−4.12E−03	**8.4057704116**	−7.18E−03	**8.4004580829**	−1.25E−02
353	8.4188352568	8.4194281857	5.93E−04	**8.4145497751**	−4.29E−03	**8.4108773467**	−7.96E−03	**8.4090274580**	−9.81E−03
354	8.4296585287	**8.4177166906**	−1.19E−02	**8.4135178730**	−1.61E−02	**8.4135178730**	−1.61E−02	**8.4135178730**	−1.61E−02
355	8.4382375759	**8.4319460786**	−6.29E−03	**8.4247274204**	−1.35E−02	**8.4239683980**	−1.43E−02	**8.4239683980**	−1.43E−02
356	8.4440909450	8.4443570488	2.66E−04	**8.4360304324**	−8.06E−03	**8.4333486287**	−1.07E−02	**8.4328444167**	−1.12E−02
357	8.4518644222	8.4540996284	2.24E−03	**8.4487831957**	−3.08E−03	**8.4453067803**	−6.56E−03	**8.4438387381**	−8.03E−03
358	8.4567949768	8.4580619333	1.27E−03	**8.4560292105**	−7.66E−04	**8.4526805450**	−4.11E−03	**8.4424537254**	−1.43E−02
359	8.4642649260	**8.4638789814**	−3.86E−04	**8.4624244276**	−1.84E−03	**8.4570407228**	−7.22E−03	**8.4570407228**	−7.22E−03
360	8.4721516869	**8.4714630109**	−6.89E−04	**8.4708687810**	−1.28E−03	**8.4656882863**	−6.46E−03	**8.4563755742**	−1.58E−02
361	8.4805536014	8.4822870635	1.73E−03	**8.4747552064**	−5.80E−03	**8.4725636473**	−7.99E−03	**8.4725636473**	−7.99E−03
362	8.4880679606	**8.4818924210**	−6.18E−03	**8.4801282417**	−7.94E−03	**8.4801282417**	−7.94E−03	**8.4801282417**	−7.94E−03
363	8.4965064991	**8.4945147585**	−1.99E−03	**8.4853235412**	−1.12E−02	**8.4850599235**	−1.14E−02	**8.4850599235**	−1.14E−02
364	8.5030156544	8.5042481035	1.23E−03	**8.5012413251**	−1.77E−03	**8.4943206659**	−8.69E−03	**8.4914009159**	−1.16E−02
365	8.5120741140	**8.5108442195**	−1.23E−03	**8.5108442195**	−1.23E−03	**8.5043586166**	−7.72E−03	**8.5007967907**	−1.13E−02
366	8.5207116927	**8.5185677895**	−2.14E−03	**8.5150859887**	−5.63E−03	**8.5112300788**	−9.48E−03	**8.5091859536**	−1.15E−02
367	8.5271184405	**8.5256457131**	−1.47E−03	**8.5243870671**	−2.73E−03	**8.5205601987**	−6.56E−03	**8.5175167006**	−9.60E−03
368	8.5351429761	**8.5347424089**	−4.01E−04	**8.5314678724**	−3.68E−03	**8.5302499922**	−4.89E−03	**8.5270427988**	−8.10E−03
369	8.5433733193	8.5457847244	2.41E−03	**8.5410350949**	−2.34E−03	**8.5351839789**	−8.19E−03	**8.5313264244**	−1.20E−02
370	8.5498955957	8.5505951920	7.00E−04	**8.5478904124**	−2.01E−03	**8.5404315001**	−9.46E−03	**8.5401310897**	−9.76E−03
371	8.5580474842	**8.5560064744**	−2.04E−03	**8.5486280037**	−9.42E−03	**8.5482896386**	−9.76E−03	**8.5342233936**	−2.38E−02
372	8.5646395490	8.5654625584	8.23E−04	**8.5623342399**	−2.31E−03	**8.5602400127**	−4.40E−03	**8.5602400127**	−4.40E−03
373	8.5718304380	8.5740746057	2.24E−03	8.5740746057	2.24E−03	**8.5660176574**	−5.81E−03	**8.5660176574**	−5.81E−03
374	8.5786934269	8.5880998853	9.41E−03	8.5821114004	3.42E−03	8.5788247276	1.31E−04	**8.5715371341**	−7.16E−03
375	8.5855784311	8.5884374413	2.86E−03	**8.5852362291**	−3.42E−04	**8.5851714255**	−4.07E−04	**8.5751239322**	−1.05E−02
376	8.5940896993	8.5969864442	2.90E−03	**8.5900055769**	−4.08E−03	**8.5865057727**	−7.58E−03	**8.5865057727**	−7.58E−03
377	8.6005441929	8.6060708648	5.53E−03	8.6043963983	3.85E−03	8.6030157264	2.47E−03	**8.5952650116**	−5.28E−03
378	8.6069492023	8.6174409939	1.05E−02	8.6111391185	4.19E−03	**8.6068106608**	−1.39E−04	**8.5993943522**	−7.55E−03
379	8.6149479886	8.6222002746	7.25E−03	8.6158922262	9.44E−04	**8.6087193733**	−6.23E−03	**8.6086773553**	−6.27E−03
380	8.6213515024	8.6243660283	3.01E−03	8.6218049375	4.53E−04	**8.6191445370**	−2.21E−03	**8.6179291061**	−3.42E−03
381	8.6290602650	8.6400455037	1.10E−02	8.6357107409	6.65E−03	8.6297937917	7.34E−04	**8.6260578765**	−3.00E−03
382	8.6388782340	8.6398611253	9.83E−04	**8.6382167273**	−6.62E−04	**8.6310089777**	−7.87E−03	**8.6309931654**	−7.89E−03
383	8.6458487389	8.6575576882	1.17E−02	8.6500970184	4.25E−03	**8.6414087792**	−4.44E−03	**8.6322034408**	−1.36E−02
384	8.6537986688	8.6599260505	6.13E−03	8.6549518407	1.15E−03	**8.6473244636**	−6.47E−03	**8.6473244403**	−6.47E−03
385	8.6597468731	8.6687784209	9.03E−03	**8.6570380866**	−2.71E−03	**8.6558895018**	−3.86E−03	**8.6558895018**	−3.86E−03
386	8.6688960219	8.6751271202	6.23E−03	**8.6675864943**	−1.31E−03	**8.6600870637**	−8.81E−03	**8.6600870637**	−8.81E−03
387	8.6743044156	8.6789051593	4.60E−03	8.6789051593	4.60E−03	**8.6682364929**	−6.07E−03	**8.6634985454**	−1.08E−02
388	8.6806940940	8.6862794418	5.59E−03	8.6841393447	3.45E−03	**8.6772501701**	−3.44E−03	**8.6738491202**	−6.84E−03
389	8.6873874968	8.6897499390	2.36E−03	**8.6850631826**	−2.32E−03	**8.6799482207**	−7.44E−03	**8.6799482207**	−7.44E−03
390	8.6956660896	8.7080886245	1.24E−02	8.7045098200	8.84E−03	**8.6934395921**	−2.23E−03	**8.6871876704**	−8.48E−03
391	8.7030232271	8.7127405787	9.72E−03	8.7092532468	6.23E−03	**8.7021824180**	−8.41E−04	**8.6970646329**	−5.96E−03
392	8.7102535414	8.7206822036	1.04E−02	8.7130898049	2.84E−03	**8.7076108328**	−2.64E−03	**8.7076108328**	−2.64E−03
393	8.7212637605	8.7247764722	3.51E−03	8.7247197734	3.46E−03	8.7220440675	7.80E−04	**8.7172997574**	−3.96E−03
394	8.7292656343	8.7319163374	2.65E−03	**8.7276780434**	−1.59E−03	**8.7253409203**	−3.92E−03	**8.7231479935**	−6.12E−03
395	8.7348094475	8.7349829757	1.74E−04	**8.7322977408**	−2.51E−03	**8.7314383339**	−3.37E−03	**8.7262329954**	−8.58E−03
396	8.7405814827	8.7536690582	1.31E−02	8.7495064161	8.92E−03	**8.7363998078**	−4.18E−03	**8.7363998078**	−4.18E−03
397	8.7466387404	8.7581396407	1.15E−02	8.7574683792	1.08E−02	8.7543810172	7.74E−03	**8.7356896248**	−1.09E−02
398	8.7534203080	8.7615899443	8.17E−03	8.7608664248	7.45E−03	**8.7490742205**	−4.35E−03	**8.7455179548**	−7.90E−03
399	8.7595077675	8.7694391180	9.93E−03	8.7688080563	9.30E−03	8.7609920714	1.48E−03	**8.7548261563**	−4.68E−03
400	8.7674798301	8.7830646218	1.56E−02	8.7786622531	1.12E−02	**8.7581904773**	−9.29E−03	**8.7581904773**	−9.29E−03
#Better		13		30		44		50	
#Equal		0		0		0		0	
#Worse		37		20		6		0	

conclusion can be confirmed in the plot. The density of the configurations obtained by SED increases as the running time increases on most of the 50 tested instances, and the density of the 50 configurations obtained in 24 h is significantly higher than the density of the best-known records. It is worth noting that the best results obtained in this experiment are different from the best results in Section 5.3 because of the randomness of our algorithm.

As discussed above, the experimental results demonstrate that the PESS problem is computationally very challenging, the difficulty of obtaining a dense configuration will increase as the number of packing items increases, and our algorithm has a very powerful solving ability on PESS instances and it can yield a better configuration with a proper and sufficient running time.

5.6. Analysis of neighbors and ANM

The neighbor structure and the proposed ANM module, described in Section 4.5, are applied to accelerate the convergence and reduce the time consumption of the continuous optimization process of the elastic PESS system. We design another experiment and present a comparison to evaluate the performance of the neighbor structure and the ANM module, described as follows.

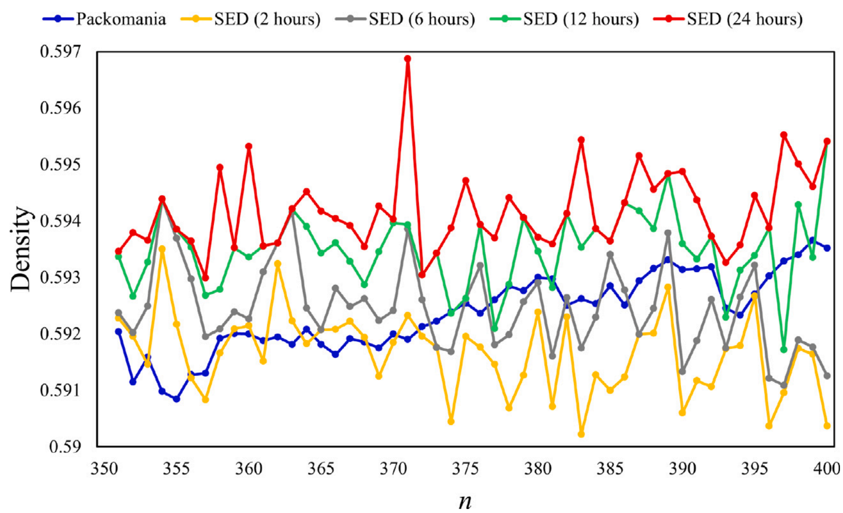
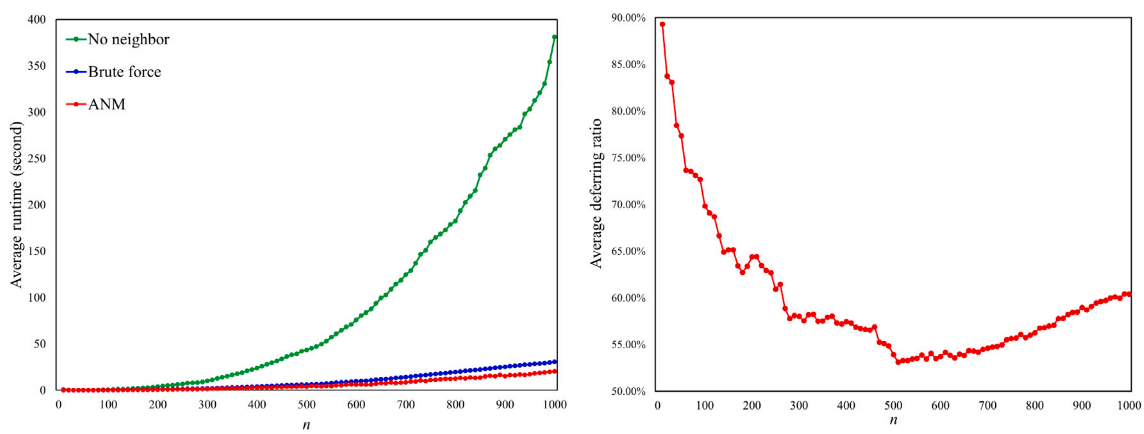


Fig. 7. Packing density plots for the best-known records from Packomania and the best results of SED with several different cut-off times on the 50 moderate-scale instances ($351 \leq n \leq 400$).



(a) Runtime comparison of brute force and ANM

(b) Deferring ratio of ANM

Fig. 8. Experimental results and comparison of the brute force approach and ANM. The left shows the average runtime plot of the two approaches and the right shows the average deferring ratio plot of ANM.

The neighbors and ANM only affect the performance of the continuous optimization process. Therefore, we make a comparison of ANM and two variants. In the first variant, the algorithm calculates the energy and gradient by enumerating all the pairs of spheres (i.e., do not use the neighbor structure), termed "No neighbor". In the second variant, the algorithm reconstructs the neighbor structure in each iteration of the continuous optimization process and applies it to calculate the energy and gradient, termed "Brute force". Starting from a random initial layout with radius $R = \sqrt[3]{\frac{n}{0.6}}$, we performed the continuous optimization algorithms of ANM and two variants until the convergence condition was met. And the experiment was performed for $n = 10, 20, 30, \ldots, 1000$, and each setting performed 1,000 independent runs. The experimental results and comparison are presented in Fig. 8.

Fig. 8(a) shows the runtime comparison of ANM and two variants, where the X-axis indicates the number of packing items and the Y-axis indicates the average runtime over 1,000 independent runs. Fig. 8(b) shows the deferring ratio plot of the ANM module, where the X-axis indicates the number of packing items and the Y-axis indicates the average deferring ratio over 1,000 independent runs, the deferring ratio is defined as $\text{Ratio}(\%) = 1 - \frac{N_{rec.}}{N_{iter.}}$, in which $N_{rec.}$ indicates the number of triggering neighbor reconstructions of the ANM module and $N_{iter.}$ indicates the number of continuous optimization iterations.

From Fig. 8, we have the following observations. The neighbor structure is a very efficient data structure of the elastic PESS system, and it can significantly reduce the time consumption of the continuous optimization process. Adopting the AMN module instead of the brute force approach can further speed up the convergence, and the gap between the brute force approach and AMN increases as the number of packing items increases. For example, the average runtimes of the brute force approach on the settings of $n = 200$, 400, 600, 800, and 1,000 are 0.96, 3.90, 9.56, 19.58, and 30.58 s, and the corresponding average runtimes of ANM are 0.66, 2.73, 6.14, 12.62, and 20.38 s where the average runtime ratio of ANM to brute force between 64.23% and 70.00%, implying that adopting the ANM module can reduce over 30% time consumption instead of the brute force approach. Besides, the ANM module defers over 50% unnecessary maintenance during the continuous optimization process for $10 \leq n \leq 1000$, as illustrated in the plot.

In experiments, we observe that the deferring ratio can be further increased by tuning the reset value and the multiplying factor of the deferring length len (refer to Algorithm 5 lines 1, 11, and 13), which can further reduce the time consumption. For example, the reset value can be fixed to 10 (i.e., using $len \leftarrow 10$ instead of $len \leftarrow 1$). In this way, ANM degenerates into a simple method (He et al., 2018), reconstructing the neighbor every 10 iterations in an unstable layout. The multiplying factor can be set to a larger value to enhance the deferring feature (i.e., using $len \leftarrow k \times len$ instead of $len \leftarrow 2 \times len$, $k > 2$). However, we do not tune these settings and hold the generality and effectiveness of our proposed ANM module.

It is worth noting that we employ the efficient k-d tree algorithm to construct the neighbor structure, gaining a low time complexity $O(n(m + n^{\frac{2}{3}}))$, in which $O(m + n^{\frac{2}{3}})$ is the complexity of range searching in a k-d tree on three-dimensional problems, and m is the number of items in the range. The gap between the brute force approach and AMN will further increase if we employ the naive construct approach (i.e., enumerating all pairs of items $O(n^2)$).

6. Conclusion

In this work, we presented an effective search algorithm to solve the Packing Equal Spheres in a Spherical container (PESS) problem, which is computationally very challenging. Two methods were proposed to achieve this goal. The first method is the Solution space Exploring and Descent (SED) heuristic, which is an efficient local search algorithm for discovering a high-quality solution based on the PESS elastic system. The second method is the Adaptive Neighbor item Maintenance (ANM) method for maintaining the efficient neighbor structure to solve packing problems, which can significantly reduce unnecessary maintenance and speed up the convergence of the continuous optimization process. The excellent performance of our algorithm was demonstrated on the well-known benchmark instances with up to 400 sphere items. Specifically, our algorithm has improved the best-known results for 274 instances and matched the best-known results for 84 instances out of the 396 benchmark instances.

On the other hand, the proposed algorithm can be further enhanced from the following directions: (1) block-coordinate descent methods can accelerate the convergence of the continuous optimization process of the PESS elastic system. These methods would be useful to tackle large-scale instances (e.g., $n \geq 1000$), and an empirically-based or geometric partition should be considered for these methods; (2) The initial solution can be constructed by a lattice arrangement, and such construction methods are more efficient than search methods. Furthermore, construction methods can also create a conflicting layout instead of packing items into the container randomly. Compared with random packing methods, construction methods can obtain a denser conflicting layout, but they are more time-consuming and might lose diversity; (3) The final result obtained in this work is quasi-feasible ($E(\boldsymbol{x}) < \epsilon$), it can apply some mathematical methods to post-process the quasi-feasible solution and obtain a truly feasible solution ($E(\boldsymbol{x}) = 0$), such as the augmented Langerian multiplier method.

Besides, our proposed SED heuristic and ANM module are of a general nature. Based on the elastic system, the SED heuristic has no restrictions on the dimension and the shape of the container of packing problems, and it can be easily adapted to other packing problems, such as circle packing problems (two-dimensional), hypersphere packing problems (high-dimensional), and problems of the container shapes for rectangle, triangle, cuboid, cylinder, etc. Meanwhile, the SED heuristic can also be adapted to solve unequal circle/sphere/hypersphere packing problems by coupling with the “swap”, “insert”, and “shift” perturbing operators, which are popularly applied in unequal item packing algorithms. The ANM module can also adapted to solve the problems mentioned above. Furthermore, because of its adaptive feature, ANM is also suitable for online packing problems. The performance of the ANM module can be further enhanced by adjusting the setting of the reset value and the multiplying factor of the deferring length depending on the characteristics and scale of the problems.

Finally, from the experimental results, we can observe that it is extremely difficult to gain a significant improvement on small-scale instances. Some discovered configurations of small-scale instances can probably be considered as the optimal solution. Instead, there is still room for improvement on moderate-scale instances.

CRediT authorship contribution statement

Jianrong Zhou: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing. **Shuo Ren:** Writing – original draft, Writing – review & editing. **Kun He:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Yanli Liu:** Resources. **Chu-Min Li:** Resources.

Data availability

Data will be made available on request.

Acknowledgements

The authors thank the reviewers for proposing insightful comments and suggestions, Dr. Eckard Specht for publishing our best solution found in this study at the Packomania website.

Appendix A. The detailed computational results and comparisons on PESS

A.1. Main experiments

Tables 8–15 summarize the detailed computational results of our algorithm on the PESS problem in the range of $5 \leq n \leq 400$. The column of n corresponds to the number of packing items in the instance. The column of R^* shows the best-known results on the Packomania website (Specht, 2023) (downloaded on March 1st, 2023). The computational results of our proposed SED algorithm are shown in columns 3 to 11. The columns of R_{best}, R_{avg}, and R_{worst} show the best, average, and worst results over 10 independent runs of SED. The columns of Δ_{best}, Δ_{avg}, and Δ_{worst} show the differences in scientific notation between R_{best}, R_{avg}, R_{worst} and R^*, i.e., $\Delta_{best} = R_{best} - R^*, \Delta_{avg} = R_{avg} - R^*, \Delta_{worst} = R_{worst} - R^*$, where a negative value indicates that SED yields a better result compared with the best-known result. The column of HR shows the success rate of hitting the best result R_{best}, and the column of RR shows the success rate of reaching or improving the best-known record. The average computation time of SED to obtain its final result is shown in the column of $Time$ (s). At the bottom of the table, the three rows of “#Better”, “#Equal”, and “#Worse” show the number of instances for which SED yields a better, equal, and worse result compared with the best-known record in terms of R_{best}, R_{avg}, and R_{worst}, respectively.

A.2. Additional experiments on different cut-off times

Tables 16–19 summarize the detailed computational results of our algorithm on the 50 moderate-scale PESS instances ($351 \leq n \leq 400$) with the four different cut-off times 2, 6, 12, and 24 h. The column of n corresponds to the number of packing items in the instance. The column of R^* shows the best-known results on the Packomania website (Specht, 2023) (downloaded on March 1st, 2023). The computational results of our proposed SED algorithm are shown in columns 3 to 11. The columns of R_{best}, R_{avg}, and R_{worst} show the best, average, and worst results over 10 independent runs of SED. The columns of Δ_{best}, Δ_{avg}, and Δ_{worst} show the differences in scientific notation between R_{best}, R_{avg}, R_{worst} and R^*, i.e., $\Delta_{best} = R_{best} - R^*, \Delta_{avg} = R_{avg} - R^*, \Delta_{worst} = R_{worst} - R^*$, where a negative value indicates that SED yields a better result compared with the best-known result. The average computation time of SED to obtain its final result is shown in the column of $Time$ (s).

Table 8
Computational results and comparison on the instances with $5 \leq n \leq 50$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
5	2.4142135624	2.4142135624	2.4142135624	2.4142135624	0	0	0	10/10	10/10	0.00
6	2.4142135624	2.4142135624	2.4142135624	2.4142135624	0	0	0	10/10	10/10	0.00
7	2.5912538723	2.5912538723	2.5912538723	2.5912538723	0	0	0	10/10	10/10	0.00
8	2.6453287760	2.6453287760	2.6453287760	2.6453287760	0	0	0	10/10	10/10	1.00
9	2.7320508076	2.7320508076	2.7320508076	2.7320508076	0	0	0	10/10	10/10	1.00
10	2.8324645611	2.8324645611	2.8324645611	2.8324645611	0	0	0	10/10	10/10	23.30
11	2.9021130326	2.9021130326	2.9021130326	2.9021130326	0	0	0	10/10	10/10	1.00
12	2.9021130326	2.9021130326	2.9021130326	2.9021130326	0	0	0	10/10	10/10	2.00
13	3.0000000000	3.0000000000	3.0000000000	3.0000000000	0	0	0	10/10	10/10	4.00
14	3.0911454449	3.0911454449	3.0911454449	3.0911454449	0	0	0	10/10	10/10	8.80
15	3.1416426249	3.1416426249	3.1416426249	3.1416426249	0	0	0	10/10	10/10	10.00
16	3.2156830320	3.2156830320	3.2156830320	3.2156830320	0	0	0	10/10	10/10	5.40
17	3.2712455117	3.2712455117	3.2712455117	3.2712455117	0	0	0	10/10	10/10	7.00
18	3.3189887817	3.3189887817	3.3189887817	3.3189887817	0	0	0	10/10	10/10	5.00
19	3.3860159733	3.3860159733	3.3860159733	3.3860159733	0	0	0	10/10	10/10	5.00
20	3.4735389622	3.4735389622	3.4735389622	3.4735389622	0	0	0	10/10	10/10	164.40
21	3.4863514104	3.4863514104	3.4863514104	3.4863514104	0	0	0	10/10	10/10	6.00
22	3.5798331912	3.5798331912	3.5798331912	3.5798331912	0	0	0	10/10	10/10	640.40
23	3.6275164365	3.6275164365	3.6275164365	3.6275164365	0	0	0	10/10	10/10	29.10
24	3.6853949355	3.6853949355	3.6853949355	3.6853949355	0	0	0	10/10	10/10	205.50
25	3.6874267475	3.6874267475	3.6874267475	3.6874267475	0	0	0	10/10	10/10	185.80
26	3.7474057765	3.7474057765	3.7474057765	3.7474057765	0	0	0	10/10	10/10	9.00
27	3.8134159569	3.8134159569	3.8134159569	3.8134159569	0	0	0	10/10	10/10	266.80
28	3.8416402781	3.8416402781	3.8416402781	3.8416402781	0	0	0	10/10	10/10	580.20
29	3.8770891032	3.8770891032	3.8770891032	3.8770891032	0	0	0	10/10	10/10	561.70
30	3.9164916616	3.9164916616	3.9164916616	3.9164916616	0	0	0	10/10	10/10	349.70
31	3.9507544849	3.9507544849	3.9507544849	3.9507544849	0	0	0	10/10	10/10	1639.20
32	3.9874403893	3.9874403893	3.9874403893	3.9874403893	0	0	0	10/10	10/10	309.90
33	4.0199009160	4.0199009160	4.0199009160	4.0199009160	0	0	0	10/10	10/10	633.20
34	4.0477199712	4.0477199712	4.0477199712	4.0477199712	0	0	0	10/10	10/10	436.90
35	4.0844057408	4.0844057408	4.0844057408	4.0844057408	0	0	0	10/10	10/10	2410.10
36	4.1129893297	4.1129893297	4.1129893297	4.1129893297	0	0	0	10/10	10/10	322.30
37	4.1547812520	4.1547812520	4.1547812520	4.1547812520	0	0	0	10/10	10/10	4287.00
38	4.1576692600	4.1576692600	4.1576692600	4.1576692600	0	0	0	10/10	10/10	7200.00
39	4.2239497563	4.2239497563	4.2239497563	4.2239497563	0	0	0	10/10	10/10	631.70
40	4.2553329537	4.2553329537	4.2553329537	4.2553329537	0	0	0	10/10	10/10	569.40
41	4.2963450048	4.2963450048	4.2963450048	4.2963450048	0	0	0	10/10	10/10	1197.90
42	4.3081420430	4.3081420430	4.3081420430	4.3081420430	0	0	0	10/10	10/10	33.00
43	4.3528798324	4.3528798324	4.3528798324	4.3528798324	0	0	0	10/10	10/10	2055.90
44	4.3828308379	4.3828308379	4.3828308379	4.3828308379	0	0	0	10/10	10/10	6145.30
45	4.4070031477	4.4070031477	4.4070031477	4.4070031477	0	0	0	10/10	10/10	1397.60
46	4.4411244747	4.4411244747	4.4411244747	4.4411244747	0	0	0	10/10	10/10	1232.30
47	4.4741318035	4.4741318035	4.4741318035	4.4741318035	0	0	0	10/10	10/10	1531.70
48	4.4962827447	4.4962827447	4.4962827447	4.4962827447	0	0	0	10/10	10/10	6323.20
49	4.5191984746	4.5191984746	4.5191984746	4.5191984746	0	0	0	10/10	10/10	2341.90
50	4.5504543407	4.5504543407	4.5504543407	4.5504543407	0	0	0	10/10	10/10	974.70
#Better		0	0	0						
#Equal		46	46	46						
#Worse		0	0	0						

At the bottom of the table, the three rows of "#Better", "#Equal", and "#Worse" show the number of instances for which SED yields a better, equal, and worse result compared with the best-known record in terms of R_{best}, R_{avg}, and R_{worst}, respectively.

Furthermore, two new indicative values are added to the table as shown in columns of $N_{iter.}$ and $N_{expl.}$ for insightful and comparable analysis purposes. $N_{iter.}$ and $N_{expl.}$ indicate the average iteration number and the average exploration number of the SED heuristic performed before the algorithm terminates over 10 runs. We call SED performing an exploration when it perturbs a local minimum solution and obtains a new local minimum solution by optimizing the perturbed solution. The iteration process corresponds to Algorithm 3 lines 5–18. The exploration process corresponds to Algorithm 3 lines 10–11.

Table 9
Computational results and comparison on the instances with $51 \leq n \leq 100$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
51	4.5756057950	4.5756057950	4.5756057950	4.5756057950	0	0	0	10/10	10/10	1518.60
52	4.6097732930	4.6097732930	4.6097732930	4.6097732930	0	0	0	10/10	10/10	6043.10
53	4.6234637833	4.6234637833	4.6234637833	4.6234637833	0	0	0	10/10	10/10	1291.10
54	4.6528795754	4.6528795754	4.6528795754	4.6528795754	0	0	0	10/10	10/10	2799.80
55	4.6851203515	4.6851203515	4.6851203515	4.6851203515	0	0	0	10/10	10/10	2504.70
56	4.6933402171	4.6933402171	4.6933402171	4.6933402171	0	0	0	10/10	10/10	7202.40
57	4.7322376049	**4.7319976099**	**4.7319976099**	**4.7319976099**	−2.40E−04	−2.40E−04	−2.40E−04	10/10	10/10	1996.00
58	4.7510429431	4.7510429431	4.7510429431	4.7510429431	0	0	0	10/10	10/10	1779.10
59	4.7673656459	4.7673656459	4.7673656459	4.7673656459	0	0	0	10/10	10/10	7202.00
60	4.7749335903	4.7749335903	4.7749335903	4.7749335903	0	0	0	10/10	10/10	922.90
61	4.7822027257	4.7822027257	4.7822027257	4.7822027257	0	0	0	10/10	10/10	1144.80
62	4.8370131185	4.8370131185	4.8370131185	4.8370131185	0	0	0	10/10	10/10	1414.60
63	4.8554314461	4.8554314461	4.8554314461	4.8554314461	0	0	0	10/10	10/10	1432.70
64	4.8993857428	4.8993857428	4.8993857428	4.8993857428	0	0	0	10/10	10/10	1685.60
65	4.9242635534	4.9242635534	4.9242635534	4.9242635534	0	0	0	10/10	10/10	7201.80
66	4.9478309581	4.9478309581	4.9478309581	4.9478309581	0	0	0	10/10	10/10	2818.70
67	4.9697680835	4.9697680835	4.9697680835	4.9697680835	0	0	0	10/10	10/10	4560.10
68	4.9999442473	4.9999442473	4.9999442473	4.9999442473	0	0	0	10/10	10/10	6503.30
69	5.0180264442	5.0180264442	5.0180264442	5.0180264442	0	0	0	10/10	10/10	2559.50
70	5.0329981188	5.0329981188	5.0329981188	5.0329981188	0	0	0	10/10	10/10	6742.30
71	5.0704246177	5.0704246177	5.0704246177	5.0704246177	0	0	0	10/10	10/10	4901.90
72	5.0946090432	5.0946090432	5.0946090432	5.0946090432	0	0	0	10/10	10/10	7202.30
73	5.1117410063	**5.1117310540**	**5.1117310540**	**5.1117310540**	−9.95E−06	−9.95E−06	−9.95E−06	10/10	10/10	7204.00
74	5.1243475815	5.1243475815	5.1243475815	5.1243475815	0	0	0	10/10	10/10	186.90
75	5.1555845175	5.1555845175	5.1555845175	5.1555845175	0	0	0	10/10	10/10	165.40
76	5.1827544049	**5.1827522689**	**5.1827522689**	**5.1827522689**	−2.14E−06	−2.14E−06	−2.14E−06	10/10	10/10	7203.50
77	5.2014502549	**5.2014494794**	**5.2014494794**	**5.2014494794**	−7.76E−07	−7.76E−07	−7.76E−07	10/10	10/10	7203.00
78	5.2230523740	5.2230523740	5.2230523740	5.2230523740	0	0	0	10/10	10/10	5712.10
79	5.2449457291	5.2449457291	5.2449457291	5.2449457291	0	0	0	10/10	10/10	2759.30
80	5.2707282625	**5.2707228597**	**5.2707230129**	**5.2707233176**	−5.40E−06	−5.25E−06	−4.94E−06	6/10	10/10	5329.10
81	5.2918921098	**5.2918269759**	**5.2918490181**	5.2919907121	−6.51E−05	−4.31E−05	9.86E−05	1/10	9/10	5812.40
82	5.3108551826	**5.3107500414**	**5.3107710697**	5.3108551826	−1.05E−04	−8.41E−05	0	8/10	10/10	4509.00
83	5.3293460126	5.3293460126	5.3295343781	5.3296101397	0	1.88E−04	2.64E−04	1/10	1/10	4292.80
84	5.3488647209	**5.3488512301**	**5.3488635868**	5.3488764897	−1.35E−05	−1.13E−06	1.18E−05	4/10	5/10	5430.70
85	5.3644833789	5.3644833789	5.3670410532	5.3695803690	0	2.56E−03	5.10E−03	3/10	3/10	5159.50
86	5.3876129492	5.3876129492	5.3876242092	5.3876930358	0	1.13E−05	8.01E−05	7/10	7/10	7003.80
87	5.4073256708	**5.4067811877**	**5.4067811877**	**5.4067811877**	−5.44E−04	−5.44E−04	−5.44E−04	10/10	10/10	4652.30
88	5.4266705368	**5.4266460069**	5.4267151869	5.4269294573	−2.45E−05	4.47E−05	2.59E−04	2/10	5/10	4285.80
89	5.4466807175	5.4466807175	5.4466807175	5.4466807175	0	0	0	10/10	10/10	3603.90
90	5.4663957267	**5.4663528574**	5.4666204162	5.4667572759	−4.29E−05	2.25E−04	3.62E−04	2/10	2/10	4929.10
91	5.4840116687	**5.4838608716**	5.4840178565	5.4841885857	−1.51E−04	6.19E−06	1.77E−04	2/10	4/10	3718.90
92	5.5010819544	5.5012326826	5.5016577070	5.5021712007	1.51E−04	5.76E−04	1.09E−03	1/10	0/10	5714.70
93	5.5167123871	**5.5161208996**	**5.5166205614**	5.5179404254	−5.91E−04	−9.18E−05	1.23E−03	1/10	8/10	3782.20
94	5.5307641333	**5.5304774705**	5.5313809950	5.5339817961	−2.87E−04	6.17E−04	3.22E−03	2/10	2/10	5457.00
95	5.5516834671	**5.5391547512**	**5.5508888816**	5.5550143011	−1.25E−02	−7.95E−04	3.33E−03	1/10	8/10	5089.50
96	5.5689451565	**5.5689435455**	**5.5689443510**	5.5689451565	−1.61E−06	−8.06E−07	0	5/10	10/10	3326.90
97	5.5859794729	5.5859794729	5.5861566571	5.5874400694	0	1.77E−04	1.46E−03	8/10	8/10	5522.60
98	5.6022822809	**5.6022558001**	5.6029733282	5.6092721967	−2.65E−05	6.91E−04	6.99E−03	3/10	9/10	5686.90
99	5.6218126153	**5.6217725678**	5.6223277124	5.6270470529	−4.00E−05	5.15E−04	5.23E−03	2/10	4/10	5889.00
100	5.6359808164	**5.6357325219**	**5.6359770664**	5.6366613233	−2.48E−04	−3.75E−06	6.81E−04	1/10	5/10	4714.50
#Better		19	13	6						
#Equal		30	26	28						
#Worse		1	11	16						

Table 10
Computational results and comparisons on the instances with $101 \leq n \leq 150$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
101	5.6599579629	**5.6597794928**	**5.6598457572**	**5.6599106765**	−1.78E−04	−1.12E−04	−4.73E−05	2/10	10/10	9941.10
102	5.6748822207	5.6748822207	5.6761187574	5.6795640862	0	1.24E−03	4.68E−03	3/10	3/10	8971.30
103	5.6923913336	**5.6921248921**	5.6928626608	5.6949214167	−2.66E−04	4.71E−04	2.53E−03	2/10	5/10	11 333.10
104	5.7089561454	**5.7082798932**	**5.7086249682**	5.7095511622	−6.76E−04	−3.31E−04	5.95E−04	1/10	8/10	11 103.20
105	5.7263005035	**5.7250734932**	**5.7251699857**	**5.7257701677**	−1.23E−03	−1.13E−03	−5.30E−04	7/10	10/10	17 688.80
106	5.7420854884	**5.7416014823**	**5.7417349458**	**5.7420322005**	−4.84E−04	−3.51E−04	−5.33E−05	5/10	10/10	14 459.00
107	5.7600955649	5.7600995060	5.7608367948	5.7614363470	3.94E−06	7.41E−04	1.34E−03	2/10	0/10	14 137.50
108	5.7747329463	**5.7747315860**	5.7753922938	5.7770470534	−1.36E−06	6.59E−04	2.31E−03	1/10	4/10	11 369.90
109	5.7924398458	5.7924398458	5.7927828765	5.7928368838	0	3.43E−04	3.97E−04	1/10	1/10	19 340.40
110	5.8043281002	5.8043790103	5.8070774340	5.8086659801	5.09E−05	2.75E−03	4.34E−03	1/10	0/10	13 714.30
111	5.8226237949	**5.8225938205**	5.8231336777	5.8253755917	−3.00E−05	5.10E−04	2.75E−03	1/10	3/10	12 636.80
112	5.8367934883	5.8381793014	5.8409569163	5.8430726193	1.39E−03	4.16E−03	6.28E−03	2/10	0/10	9841.40
113	5.8529505194	5.8529505194	5.8537121832	5.8574056527	0	7.62E−04	4.46E−03	1/10	1/10	10 367.80
114	5.8668408971	**5.8668377535**	5.8711734848	5.8748666977	−3.14E−06	4.33E−03	8.03E−03	1/10	1/10	7843.10
115	5.8911773387	**5.8877955184**	**5.8890441963**	5.8923746687	−3.38E−03	−2.13E−03	1.20E−03	1/10	8/10	8656.50
116	5.9043232575	**5.9026298032**	5.9045535851	5.9074851958	−1.69E−03	2.30E−04	3.16E−03	1/10	4/10	9062.60
117	5.9151425496	**5.9151356434**	5.9155961822	5.9190045152	−6.91E−06	4.54E−04	3.86E−03	1/10	1/10	9359.70
118	5.9268531543	5.9268531543	5.9268531543	5.9268531543	0	0	0	10/10	10/10	7391.40
119	5.9491701550	**5.9490619877**	**5.9491643600**	5.9494410044	−1.08E−04	−5.79E−06	2.71E−04	4/10	8/10	11 968.90
120	5.9668074819	**5.9636279898**	**5.9645070091**	5.9669104091	−3.18E−03	−2.30E−03	1.03E−04	1/10	9/10	12 676.80
121	5.9827857339	**5.9802333759**	**5.9811283874**	5.9828268525	−2.55E−03	−1.66E−03	4.11E−05	1/10	9/10	11 757.50
122	5.9960642820	**5.9936598394**	**5.9947179557**	**5.9957195013**	−2.40E−03	−1.35E−03	−3.45E−04	1/10	10/10	7227.90
123	6.0090785558	**6.0059664994**	**6.0063563277**	**6.0072839085**	−3.11E−03	−2.72E−03	−1.79E−03	4/10	10/10	5469.00
124	6.0225351062	**6.0224402824**	6.0231508009	6.0250308876	−9.48E−05	6.16E−04	2.50E−03	1/10	3/10	9234.90
125	6.0380168643	**6.0347337708**	**6.0356903597**	**6.0378742514**	−3.28E−03	−2.33E−03	−1.43E−04	1/10	10/10	8789.40
126	6.0521543613	**6.0519428829**	6.0523238658	6.0528280918	−2.11E−04	1.70E−04	6.74E−04	1/10	2/10	14 472.60
127	6.0668536575	**6.0665232674**	6.0669545910	6.0680581019	−3.30E−04	1.01E−04	1.20E−03	3/10	6/10	14 369.10
128	6.0806366813	**6.0797285101**	6.0807445392	6.0825703191	−9.08E−04	1.08E−04	1.93E−03	1/10	6/10	10 502.80
129	6.0932666869	**6.0922757381**	**6.0930011686**	6.0948755179	−9.91E−04	−2.66E−04	1.61E−03	2/10	7/10	10 997.50
130	6.1076531690	**6.1072029065**	6.1076651361	6.1083061457	−4.50E−04	1.20E−05	6.53E−04	1/10	4/10	14 599.50
131	6.1178856994	**6.1146523017**	**6.1157575742**	6.1191827681	−3.23E−03	−2.13E−03	1.30E−03	1/10	8/10	4180.80
132	6.1269393558	**6.1262395723**	**6.1267578890**	6.1271243441	−7.00E−04	−1.81E−04	1.85E−04	1/10	8/10	16 765.70
133	6.1363940823	**6.1363253673**	**6.1363643585**	6.1365124399	−6.87E−05	−2.97E−05	1.18E−04	6/10	7/10	19 921.60
134	6.1412839494	**6.1412827373**	6.1413091994	6.1415468235	−1.21E−06	2.53E−05	2.63E−04	5/10	9/10	19 320.70
135	6.1508293815	6.1508293815	6.1508295653	6.1508300088	0	1.84E−07	6.27E−07	3/10	3/10	21 618.80
136	6.1584969194	**6.1584884950**	**6.1584894041**	**6.1584899011**	−8.42E−06	−7.52E−06	−7.02E−06	2/10	10/10	17 476.90
137	6.1815082562	6.1815082562	6.1815082562	6.1815082562	0	0	0	10/10	10/10	16 313.80
138	6.2059663809	**6.2059641354**	6.2060674407	6.2064484742	−2.25E−06	1.01E−04	4.82E−04	6/10	7/10	19 133.70
139	6.2210255874	**6.2209771473**	6.2210550401	6.2211316778	−4.84E−05	2.95E−05	1.06E−04	1/10	4/10	21 004.10
140	6.2341312659	**6.2341210439**	6.2342623742	6.2347871644	−1.02E−05	1.31E−04	6.56E−04	6/10	7/10	21 423.80
141	6.2490415957	**6.2483731721**	**6.2483929733**	**6.2485423294**	−6.68E−04	−6.49E−04	−4.99E−04	3/10	10/10	21 345.70
142	6.2617416866	6.2617416866	6.2617448548	6.2617483397	0	3.17E−06	6.65E−06	4/10	4/10	20 381.20
143	6.2716966903	**6.2715956530**	**6.2715957051**	**6.2715957605**	−1.01E−04	−1.01E−04	−1.01E−04	3/10	10/10	21 629.90
144	6.2832371987	**6.2832364663**	**6.2832370480**	**6.2832371934**	−7.32E−07	−1.51E−07	−5.30E−09	2/10	10/10	21 109.70
145	6.3051401291	**6.3048057891**	**6.3048623098**	**6.3049160579**	−3.34E−04	−2.78E−04	−2.24E−04	4/10	10/10	19 513.10
146	6.3192938478	**6.3191616112**	**6.3191867233**	6.3193217626	−1.32E−04	−1.07E−04	2.79E−05	7/10	9/10	19 888.90
147	6.3377734986	6.3377734986	6.3377957984	6.3379502687	0	2.23E−05	1.77E−04	6/10	6/10	20 428.90
148	6.3535817257	**6.3465377175**	**6.3515717482**	**6.3528809630**	−7.04E−03	−2.01E−03	−7.01E−04	1/10	10/10	15 163.10
149	6.3663869913	**6.3654709825**	**6.3663756849**	6.3673058153	−9.16E−04	−1.13E−05	9.19E−04	1/10	4/10	15 256.50
150	6.3814711453	**6.3800749408**	**6.3809350553**	**6.3813306116**	−1.40E−03	−5.36E−04	−1.41E−04	1/10	10/10	14 089.80
#Better		39	24	13						
#Equal		8	2	2						
#Worse		3	24	35						

Table 11
Computational results and comparisons on the instances with $151 \le n \le 200$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
151	6.3985999459	**6.3840816057**	**6.3896769602**	**6.3984254479**	−1.45E−02	−8.92E−03	−1.74E−04	1/10	10/10	16 216.80
152	6.4096321832	**6.4084860226**	6.4097374079	6.4117044502	−1.15E−03	1.05E−04	2.07E−03	2/10	5/10	10 276.00
153	6.4275041177	**6.4229011359**	**6.4236943663**	**6.4256045014**	−4.60E−03	−3.81E−03	−1.90E−03	1/10	10/10	13 533.70
154	6.4419628019	**6.4363632879**	**6.4392664417**	**6.4408574212**	−5.60E−03	−2.70E−03	−1.11E−03	1/10	10/10	8635.50
155	6.4564928744	**6.4537179487**	**6.4546576387**	6.4567013952	−2.77E−03	−1.84E−03	2.09E−04	2/10	9/10	11 761.80
156	6.4687030989	**6.4664665398**	**6.4666929851**	6.4687309932	−2.24E−03	−2.01E−03	2.79E−05	9/10	9/10	5981.10
157	6.4806879723	**6.4793187351**	**6.4801269581**	6.4814134832	−1.37E−03	−5.61E−04	7.26E−04	1/10	8/10	7711.40
158	6.4901644711	**6.4899861268**	6.4906221569	6.4931288714	−1.78E−04	4.58E−04	2.96E−03	1/10	5/10	9042.90
159	6.5063519438	**6.5050243930**	**6.5054759138**	**6.5061752607**	−1.33E−03	−8.76E−04	−1.77E−04	1/10	10/10	7050.70
160	6.5216146623	**6.5187621450**	**6.5203667820**	6.5218506654	−2.85E−03	−1.25E−03	2.36E−04	1/10	9/10	9556.50
161	6.5318675905	**6.5315929690**	6.5330487499	6.5348329328	−2.75E−04	1.18E−03	2.97E−03	1/10	4/10	11 714.50
162	6.5459496371	**6.5450672093**	6.5460087564	6.5464123346	−8.82E−04	5.91E−05	4.63E−04	3/10	3/10	5743.30
163	6.5645506055	**6.5577283543**	**6.5584221542**	**6.5596897108**	−6.82E−03	−6.13E−03	−4.86E−03	1/10	10/10	7448.80
164	6.5747462297	**6.5695683516**	**6.5697443165**	**6.5703998385**	−5.18E−03	−5.00E−03	−4.35E−03	2/10	10/10	15 182.80
165	6.5868295847	**6.5794341232**	**6.5795512291**	**6.5799560441**	−7.40E−03	−7.28E−03	−6.87E−03	1/10	10/10	14 182.70
166	6.6002228950	**6.5951677305**	**6.5958877885**	**6.5982060561**	−5.06E−03	−4.34E−03	−2.02E−03	1/10	10/10	12 318.40
167	6.6104638217	**6.6098597108**	**6.6102879842**	6.6105208897	−6.04E−04	−1.76E−04	5.71E−05	1/10	5/10	7921.20
168	6.6242823896	**6.6230917029**	**6.6231154345**	**6.6233215699**	−1.19E−03	−1.17E−03	−9.61E−04	2/10	10/10	4024.90
169	6.6386257645	**6.6363389670**	**6.6372393674**	**6.6380966353**	−2.29E−03	−1.39E−03	−5.29E−04	1/10	10/10	12 286.80
170	6.6482796492	**6.6466042999**	6.6489536764	6.6494813742	−1.68E−03	6.74E−04	1.20E−03	1/10	2/10	9279.90
171	6.6610392083	**6.6606281796**	6.6619873814	6.6647096408	−4.11E−04	9.48E−04	3.67E−03	2/10	3/10	10 573.10
172	6.6745006321	**6.6714601988**	**6.6732211904**	6.6772090399	−3.04E−03	−1.28E−03	2.71E−03	1/10	8/10	10 023.20
173	6.6866891682	**6.6818155183**	**6.6837432870**	**6.6855588718**	−4.87E−03	−2.95E−03	−1.13E−03	1/10	10/10	10 865.60
174	6.6981105869	**6.6973820953**	6.6983253058	6.7001634902	−7.28E−04	2.15E−04	2.05E−03	1/10	4/10	7223.20
175	6.7110188626	**6.7101346800**	6.7111654896	6.7127328028	−8.84E−04	1.47E−04	1.71E−03	1/10	4/10	12 185.90
176	6.7228800450	**6.7228629814**	6.7245784136	6.7273511525	−1.71E−05	1.70E−03	4.47E−03	2/10	2/10	8947.00
177	6.7412137384	**6.7374499882**	**6.7390210504**	**6.7397702544**	−3.76E−03	−2.19E−03	−1.44E−03	1/10	10/10	9173.10
178	6.7537260778	**6.7500785679**	**6.7519579397**	**6.7534555051**	−3.65E−03	−1.77E−03	−2.71E−04	1/10	10/10	10 426.40
179	6.7668261653	**6.7629223995**	**6.7637985814**	**6.7658683762**	−3.90E−03	−3.03E−03	−9.58E−04	1/10	10/10	8457.30
180	6.7728585654	6.7729312241	6.7753088647	6.7775098532	7.27E−05	2.45E−03	4.65E−03	1/10	0/10	8947.10
181	6.7924484211	**6.7866497775**	**6.7889594255**	**6.7914705413**	−5.80E−03	−3.49E−03	−9.78E−04	1/10	10/10	7381.40
182	6.8047492229	**6.7999756316**	**6.8017976377**	**6.8044553575**	−4.77E−03	−2.95E−03	−2.94E−04	1/10	10/10	11 125.60
183	6.8160062463	**6.8129617420**	**6.8141746128**	**6.8158045042**	−3.04E−03	−1.83E−03	−2.02E−04	1/10	10/10	5952.90
184	6.8283189424	**6.8249182583**	**6.8276016309**	6.8314937557	−3.40E−03	−7.17E−04	3.17E−03	1/10	9/10	4433.20
185	6.8424112657	**6.8374332956**	**6.8396361609**	**6.8423043887**	−4.98E−03	−2.78E−03	−1.07E−04	1/10	10/10	5675.60
186	6.8552427473	**6.8504870996**	**6.8532734673**	6.8562630968	−4.76E−03	−1.97E−03	1.02E−03	1/10	9/10	10 833.40
187	6.8661850359	**6.8631014993**	**6.8657471912**	6.8682882847	−3.08E−03	−4.38E−04	2.10E−03	1/10	5/10	11 367.00
188	6.8783760115	**6.8753773078**	**6.8776170850**	6.8802933680	−3.00E−03	−7.59E−04	1.92E−03	1/10	7/10	9281.30
189	6.8869865649	**6.8853029301**	6.8898744153	6.8938162573	−1.68E−03	2.89E−03	6.83E−03	1/10	3/10	11 578.30
190	6.8992268282	6.9009579733	6.9027526865	6.9067048144	1.73E−03	3.53E−03	7.48E−03	1/10	0/10	15 317.90
191	6.9119625946	**6.9104619032**	6.9136540117	6.9171428936	−1.50E−03	1.69E−03	5.18E−03	1/10	2/10	12 324.30
192	6.9229251057	**6.9203258423**	6.9236192514	6.9273422989	−2.60E−03	6.94E−04	4.42E−03	1/10	4/10	14 234.80
193	6.9364631911	**6.9329148694**	6.9370827791	6.9417511986	−3.55E−03	6.20E−04	5.29E−03	1/10	4/10	7599.70
194	6.9506991452	**6.9405887493**	**6.9481145753**	6.9517340557	−1.01E−02	−2.58E−03	1.03E−03	1/10	8/10	11 619.90
195	6.9631085477	**6.9551013467**	**6.9603931174**	6.9645900599	−8.01E−03	−2.72E−03	1.48E−03	1/10	7/10	12 510.10
196	6.9772480051	**6.9651197544**	**6.9704765550**	**6.9769144451**	−1.21E−02	−6.77E−03	−3.34E−04	1/10	10/10	8656.90
197	6.9878958414	**6.9790672127**	**6.9832654056**	**6.9865851286**	−8.83E−03	−4.63E−03	−1.31E−03	1/10	10/10	12 163.60
198	6.9979276863	**6.9892007374**	**6.9949396268**	6.9984685341	−8.73E−03	−2.99E−03	5.41E−04	1/10	9/10	12 900.40
199	7.0101428170	**7.0033775577**	**7.0065604316**	**7.0098414125**	−6.77E−03	−3.58E−03	−3.01E−04	1/10	9/10	17 047.00
200	7.0226524557	**7.0160475056**	**7.0199497597**	**7.0220139278**	−6.60E−03	−2.70E−03	−6.39E−04	1/10	10/10	20 248.00
#Better		48	35	22						
#Equal		0	0	0						
#Worse		2	15	28						

Table 12
Computational results and comparisons on the instances with $201 \le n \le 250$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
201	7.0311579735	**7.0255544330**	**7.0291874305**	7.0348004337	−5.60E−03	−1.97E−03	3.64E−03	1/10	8/10	21 949.40
202	7.0452775617	**7.0358154477**	**7.0400322271**	**7.0438676649**	−9.46E−03	−5.25E−03	−1.41E−03	1/10	10/10	27 886.50
203	7.0547724371	**7.0484216078**	**7.0511581917**	**7.0545886081**	−6.35E−03	−3.61E−03	−1.84E−04	1/10	10/10	25 251.60
204	7.0654812476	**7.0590273102**	**7.0617579008**	7.0660277184	−6.45E−03	−3.72E−03	5.46E−04	1/10	9/10	24 885.50
205	7.0793921434	**7.0632644486**	**7.0702898675**	**7.0769124763**	−1.61E−02	−9.10E−03	−2.48E−03	1/10	10/10	18 411.00
206	7.0896011259	**7.0812834485**	**7.0855049261**	**7.0883834243**	−8.32E−03	−4.10E−03	−1.22E−03	1/10	10/10	20 135.40
207	7.1003436731	**7.0909103971**	**7.0945016181**	**7.0993242907**	−9.43E−03	−5.84E−03	−1.02E−03	1/10	10/10	29 044.60
208	7.1114371819	**7.1043854402**	**7.1072611271**	**7.1089008345**	−7.05E−03	−4.18E−03	−2.54E−03	1/10	10/10	29 433.20
209	7.1225774947	**7.1158587077**	**7.1192283319**	**7.1223341136**	−6.72E−03	−3.35E−03	−2.43E−04	1/10	10/10	29 498.50
210	7.1340172788	**7.1249463415**	**7.1287760840**	**7.1305274090**	−9.07E−03	−5.24E−03	−3.49E−03	1/10	10/10	28 973.70
211	7.1453393266	**7.1353927996**	**7.1402314896**	**7.1430682717**	−9.95E−03	−5.11E−03	−2.27E−03	1/10	10/10	24 398.70
212	7.1571882237	**7.1474588978**	**7.1490794335**	**7.1518362056**	−9.73E−03	−8.11E−03	−5.35E−03	1/10	10/10	24 013.30
213	7.1672317136	**7.1573496074**	**7.1604642324**	**7.1639970006**	−9.88E−03	−6.77E−03	−3.23E−03	1/10	10/10	24 727.20
214	7.1767228307	**7.1668129415**	**7.1696479254**	**7.1723277016**	−9.91E−03	−7.07E−03	−4.40E−03	1/10	10/10	31 065.30
215	7.1858107754	**7.1779075067**	**7.1808898146**	**7.1837968662**	−7.90E−03	−4.92E−03	−2.01E−03	1/10	10/10	30 391.20
216	7.1962064445	**7.1876566450**	**7.1930848655**	**7.1957175098**	−8.55E−03	−3.12E−03	−4.89E−04	1/10	10/10	29 824.10
217	7.2037038757	**7.1977035204**	**7.2018444199**	7.2049872434	−6.00E−03	−1.86E−03	1.28E−03	1/10	8/10	29 494.90
218	7.2119969745	**7.2089282580**	7.2138131185	7.2178203366	−3.07E−03	1.82E−03	5.82E−03	1/10	2/10	28 066.90
219	7.2248960523	**7.2203773197**	**7.2230900357**	7.2252997380	−4.52E−03	−1.81E−03	4.04E−04	1/10	8/10	33 260.30
220	7.2365800516	**7.2296491612**	**7.2347172798**	7.2387679709	−6.93E−03	−1.86E−03	2.19E−03	1/10	8/10	21 758.60
221	7.2462659092	**7.2412540521**	**7.2435271826**	7.2467843632	−5.01E−03	−2.74E−03	5.18E−04	1/10	9/10	27 656.20
222	7.2551034048	**7.2500633764**	**7.2527363680**	**7.2548814616**	−5.04E−03	−2.37E−03	−2.22E−04	1/10	10/10	34 211.40
223	7.2656309049	**7.2580609579**	**7.2627742904**	7.2671715615	−7.57E−03	−2.86E−03	1.54E−03	1/10	9/10	29 684.30
224	7.2773063486	**7.2696697784**	**7.2763985458**	7.2815755290	−7.64E−03	−9.08E−04	4.27E−03	1/10	5/10	17 439.50
225	7.2841046441	**7.2813658072**	7.2850444400	7.2901163132	−2.74E−03	9.40E−04	6.01E−03	1/10	4/10	25 900.90
226	7.2923021211	**7.2868642085**	7.2929506330	7.2962592357	−5.44E−03	6.49E−04	3.96E−03	1/10	3/10	34 965.30
227	7.3067387277	**7.3013688606**	**7.3035699495**	**7.3060234837**	−5.37E−03	−3.17E−03	−7.15E−04	1/10	10/10	26 158.50
228	7.3163529791	**7.3086523632**	**7.3128750428**	**7.3159810279**	−7.70E−03	−3.48E−03	−3.72E−04	1/10	10/10	30 731.00
229	7.3304940328	**7.3230948777**	**7.3244515439**	**7.3270906185**	−7.40E−03	−6.04E−03	−3.40E−03	1/10	10/10	28 738.70
230	7.3387673249	**7.3304753817**	**7.3323861104**	**7.3358691984**	−8.29E−03	−6.38E−03	−2.90E−03	1/10	10/10	31 189.50
231	7.3478959531	**7.3395741975**	**7.3433421654**	**7.3458864889**	−8.32E−03	−4.55E−03	−2.01E−03	1/10	10/10	26 296.50
232	7.3606440021	**7.3514522524**	**7.3538407475**	**7.3575767617**	−9.19E−03	−6.80E−03	−3.07E−03	1/10	10/10	29 602.60
233	7.3660814854	**7.3597164787**	**7.3633262498**	7.3681790896	−6.37E−03	−2.76E−03	2.10E−03	1/10	8/10	24 688.20
234	7.3783260883	**7.3706429647**	**7.3737163256**	**7.3771612367**	−7.68E−03	−4.61E−03	−1.16E−03	1/10	10/10	22 580.70
235	7.3870071716	**7.3818305285**	**7.3847166386**	7.3872149138	−5.18E−03	−2.29E−03	2.08E−04	1/10	9/10	26 662.10
236	7.3941962819	**7.3915014587**	7.3961418565	7.3992044818	−2.69E−03	1.95E−03	5.01E−03	1/10	2/10	24 757.80
237	7.4033253417	**7.3993046206**	7.4056526724	7.4085958418	−4.02E−03	2.33E−03	5.27E−03	1/10	2/10	26 060.40
238	7.4151037322	**7.4144807701**	7.4181620386	7.4220090877	−6.23E−04	3.06E−03	6.91E−03	1/10	2/10	32 024.30
239	7.4305286960	**7.4202453485**	**7.4256091484**	7.4312513241	−1.03E−02	−4.92E−03	7.23E−04	1/10	10/10	37 016.80
240	7.4396535786	**7.4295984609**	**7.4369271123**	7.4406337248	−1.01E−02	−2.73E−03	9.80E−04	1/10	8/10	34 259.00
241	7.4493181639	**7.4352855259**	**7.4436434751**	**7.4485920810**	−1.40E−02	−5.67E−03	−7.26E−04	1/10	10/10	32 379.40
242	7.4583263208	**7.4483780897**	**7.4533697575**	7.4587836021	−9.95E−03	−4.96E−03	4.57E−04	1/10	9/10	33 125.20
243	7.4662779719	**7.4543237759**	**7.4601566836**	**7.4658048723**	−1.20E−02	−6.12E−03	−4.73E−04	1/10	10/10	34 035.40
244	7.4755562089	**7.4638668575**	**7.4693595567**	7.4759916718	−1.17E−02	−6.20E−03	4.35E−04	1/10	9/10	40 302.80
245	7.4855419380	**7.4754936428**	**7.4787240443**	**7.4848462810**	−1.00E−02	−6.82E−03	−6.96E−04	1/10	10/10	28 282.10
246	7.4908312394	**7.4833047976**	**7.4887762093**	7.4929495601	−7.53E−03	−2.06E−03	2.12E−03	1/10	6/10	26 785.80
247	7.4975449341	**7.4923815998**	**7.4954534948**	7.5049336983	−5.16E−03	−2.09E−03	7.39E−03	1/10	7/10	38 049.30
248	7.5072043746	**7.4995291317**	**7.5059041739**	7.5100113921	−7.68E−03	−1.30E−03	2.81E−03	1/10	5/10	17 319.50
249	7.5203328104	**7.5092769099**	**7.5117939937**	7.5215633878	−1.11E−02	−8.54E−03	1.23E−03	1/10	9/10	18 143.60
250	7.5270535066	**7.5177181629**	**7.5196028680**	7.5288136313	−9.34E−03	−7.45E−03	1.76E−03	1/10	9/10	25 389.90
#Better		50	44	25						
#Equal		0	0	0						
#Worse		0	6	25						

Table 13
Computational results and comparisons on the instances with $251 \le n \le 300$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
251	7.5340290155	**7.5263662601**	**7.5272731866**	**7.5313960738**	−7.66E−03	−6.76E−03	−2.63E−03	1/10	10/10	12 987.90
252	7.5424648986	**7.5340412095**	**7.5343693954**	**7.5361665730**	−8.42E−03	−8.10E−03	−6.30E−03	1/10	10/10	24 414.60
253	7.5521308100	**7.5424247553**	**7.5426571038**	**7.5430519604**	−9.71E−03	−9.47E−03	−9.08E−03	6/10	10/10	26 944.60
254	7.5621077128	**7.5482890171**	**7.5495184489**	**7.5590073907**	−1.38E−02	−1.26E−02	−3.10E−03	2/10	10/10	19 186.80
255	7.5718630254	**7.5633379207**	**7.5647885091**	**7.5686061680**	−8.53E−03	−7.07E−03	−3.26E−03	3/10	10/10	21 495.70
256	7.5793149724	**7.5721483921**	**7.5751185507**	7.5831752356	−7.17E−03	−4.20E−03	3.86E−03	1/10	8/10	23 250.50
257	7.5923599786	**7.5806235741**	**7.5823156930**	**7.5894349229**	−1.17E−02	−1.00E−02	−2.93E−03	1/10	10/10	26 719.70
258	7.6081822216	**7.5937072918**	**7.5963308481**	**7.6029758879**	−1.45E−02	−1.19E−02	−5.21E−03	1/10	10/10	22 982.20
259	7.6232794657	**7.6054822855**	**7.6071154633**	**7.6153703849**	−1.78E−02	−1.62E−02	−7.91E−03	1/10	10/10	25 211.30
260	7.6303474460	**7.6147896386**	**7.6204840351**	**7.6275037021**	−1.56E−02	−9.86E−03	−2.84E−03	1/10	10/10	29 490.10
261	7.6389568630	**7.6249382747**	**7.6309926787**	7.6399442834	−1.40E−02	−7.96E−03	9.87E−04	1/10	9/10	28 613.60
262	7.6485119939	**7.6350999584**	**7.6405314425**	7.6513694414	−1.34E−02	−7.98E−03	2.86E−03	1/10	9/10	26 678.30
263	7.6627443277	**7.6430467452**	**7.6509048689**	**7.6555794329**	−1.97E−02	−1.18E−02	−7.16E−03	1/10	10/10	35 999.60
264	7.6713523051	**7.6609426956**	**7.6651209911**	**7.6699861414**	−1.04E−02	−6.23E−03	−1.37E−03	1/10	10/10	32 310.60
265	7.6835789969	**7.6668328972**	**7.6727525059**	**7.6813143971**	−1.67E−02	−1.08E−02	−2.26E−03	1/10	10/10	34 413.50
266	7.6903773179	**7.6776988774**	**7.6825349053**	**7.6872074543**	−1.27E−02	−7.84E−03	−3.17E−03	1/10	10/10	37 073.50
267	7.6944927612	**7.6860055622**	**7.6922616494**	7.6970168113	−8.49E−03	−2.23E−03	2.52E−03	1/10	8/10	37 946.30
268	7.6988832962	**7.6964397347**	7.7030194411	7.7052583180	−2.44E−03	4.14E−03	6.38E−03	1/10	1/10	33 625.30
269	7.7044148510	7.7048369985	7.7101553679	7.7156308999	4.22E−04	5.74E−03	1.12E−02	1/10	0/10	38 941.10
270	7.7082910864	7.7112854992	7.7191886111	7.7270770927	2.99E−03	1.09E−02	1.88E−02	1/10	0/10	34 779.00
271	7.7123271806	7.7202832347	7.7291912698	7.7342023765	7.96E−03	1.69E−02	2.19E−02	1/10	0/10	39 066.90
272	7.7217513975	7.7333575768	7.7384232551	7.7429300491	1.16E−02	1.67E−02	2.12E−02	1/10	0/10	37 883.30
273	7.7283368304	7.7411887098	7.7465249044	7.7527816534	1.29E−02	1.82E−02	2.44E−02	1/10	0/10	38 376.30
274	7.7386433178	7.7493245537	7.7550211192	7.7601538373	1.07E−02	1.64E−02	2.15E−02	1/10	0/10	29 294.40
275	7.7472814344	7.7574168452	7.7646436754	7.7697877541	1.01E−02	1.74E−02	2.25E−02	1/10	0/10	35 441.30
276	7.7633757754	**7.7537790480**	7.7736485773	7.7826413059	−9.60E−03	1.03E−02	1.93E−02	1/10	1/10	36 941.20
277	7.7685794251	7.7711489967	7.7806041894	7.7864391742	2.57E−03	1.20E−02	1.79E−02	1/10	0/10	36 317.00
278	7.7753213557	**7.7705949667**	7.7889657025	7.7952769364	−4.73E−03	1.36E−02	2.00E−02	1/10	1/10	34 971.70
279	7.7828242480	7.7925896802	7.7985418673	7.8035448293	9.77E−03	1.57E−02	2.07E−02	1/10	0/10	32 826.30
280	7.7895974859	**7.7850390707**	7.8009574613	7.8127166041	−4.56E−03	1.14E−02	2.31E−02	1/10	4/10	34 835.80
281	7.7970295470	**7.7935255540**	7.8046367242	7.8172666334	−3.50E−03	7.61E−03	2.02E−02	1/10	4/10	27 388.20
282	7.8094475872	**7.8025455856**	7.8136332483	7.8263410488	−6.90E−03	4.19E−03	1.69E−02	1/10	5/10	28 618.10
283	7.8169572685	**7.8094014911**	7.8241488493	7.8380078812	−7.56E−03	7.19E−03	2.11E−02	1/10	4/10	23 766.10
284	7.8239824233	**7.8157677410**	7.8256414787	7.8407195186	−8.21E−03	1.66E−03	1.67E−02	1/10	6/10	29 747.60
285	7.8308005585	**7.8251346857**	7.8311060285	7.8540795310	−5.67E−03	3.05E−04	2.33E−02	4/10	8/10	30 489.20
286	7.8409629540	**7.8335735779**	**7.8399883454**	7.8600199617	−7.39E−03	−9.75E−04	1.91E−02	1/10	8/10	28 673.30
287	7.8498687648	**7.8428652701**	**7.8441472545**	**7.8455731732**	−7.00E−03	−5.72E−03	−4.30E−03	1/10	10/10	32 973.60
288	7.8614608232	**7.8508039705**	**7.8521664822**	**7.8563266950**	−1.07E−02	−9.29E−03	−5.13E−03	1/10	10/10	29 031.00
289	7.8700993136	**7.8605264864**	**7.8626203894**	**7.8640637814**	−9.57E−03	−7.48E−03	−6.04E−03	1/10	10/10	31 146.20
290	7.8790360850	**7.8698725201**	**7.8715351972**	**7.8729073885**	−9.16E−03	−7.50E−03	−6.13E−03	1/10	10/10	32 901.30
291	7.8887911114	**7.8793971860**	**7.8800645151**	**7.8818949248**	−9.39E−03	−8.73E−03	−6.90E−03	1/10	10/10	31 435.40
292	7.8973927600	**7.8877228593**	**7.8888227597**	**7.8906823188**	−9.67E−03	−8.57E−03	−6.71E−03	1/10	10/10	31 993.50
293	7.9052966809	**7.8963644352**	**7.8982809450**	**7.9027151560**	−8.93E−03	−7.02E−03	−2.58E−03	1/10	10/10	39 189.50
294	7.9140909168	**7.9055450950**	**7.9060972510**	**7.9067960760**	−8.55E−03	−7.99E−03	−7.29E−03	1/10	10/10	34 029.60
295	7.9215067329	**7.9125134312**	**7.9133814366**	**7.9196431482**	−8.99E−03	−8.13E−03	−1.86E−03	1/10	10/10	36 254.10
296	7.9301910270	**7.9218499738**	**7.9221697952**	**7.9230442428**	−8.34E−03	−8.02E−03	−7.15E−03	1/10	10/10	33 714.60
297	7.9439217715	**7.9296931750**	**7.9306822749**	**7.9381451750**	−1.42E−02	−1.32E−02	−5.78E−03	1/10	10/10	34 350.10
298	7.9562684663	**7.9392349019**	**7.9398510754**	**7.9406007441**	−1.70E−02	−1.64E−02	−1.57E−02	1/10	10/10	36 449.80
299	7.9615077588	**7.9476166544**	**7.9491620619**	**7.9507824188**	−1.39E−02	−1.23E−02	−1.07E−02	1/10	10/10	35 998.00
300	7.9716783431	**7.9589157181**	**7.9604790441**	**7.9640760609**	−1.28E−02	−1.12E−02	−7.60E−03	1/10	10/10	37 021.50
#Better		41	32	27						
#Equal		0	0	0						
#Worse		9	18	23						

Table 14
Computational results and comparisons on the instances with $301 \le n \le 350$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
301	7.9811692972	**7.9688310111**	**7.9703173229**	**7.9751359887**	−1.23E−02	−1.09E−02	−6.03E−03	1/10	10/10	31 465.50
302	7.9916010741	**7.9761636886**	**7.9780148165**	**7.9843134619**	−1.54E−02	−1.36E−02	−7.29E−03	1/10	10/10	30 879.40
303	7.9983093317	**7.9871618864**	**7.9891388900**	**7.9920698222**	−1.11E−02	−9.17E−03	−6.24E−03	1/10	10/10	32 947.10
304	8.0074253659	**7.9947232300**	**7.9977950015**	**8.0016672540**	−1.27E−02	−9.63E−03	−5.76E−03	1/10	10/10	35 196.30
305	8.0131815877	**8.0046510979**	**8.0066796544**	**8.0106413628**	−8.53E−03	−6.50E−03	−2.54E−03	1/10	10/10	31 234.90
306	8.0213078802	**8.0124605464**	**8.0161690264**	**8.0196765351**	−8.85E−03	−5.14E−03	−1.63E−03	1/10	10/10	29 711.90
307	8.0301086572	**8.0226883734**	**8.0247586057**	**8.0268988653**	−7.42E−03	−5.35E−03	−3.21E−03	1/10	10/10	31 324.90
308	8.0380891717	**8.0321834308**	**8.0337171490**	**8.0359984134**	−5.91E−03	−4.37E−03	−2.09E−03	1/10	10/10	32 555.50
309	8.0448618089	**8.0411618615**	**8.0441601989**	8.0518841721	−3.70E−03	−7.02E−04	7.02E−03	1/10	6/10	29 683.10
310	8.0548522945	**8.0464675478**	**8.0496913899**	**8.0520362586**	−8.38E−03	−5.16E−03	−2.82E−03	1/10	10/10	34 729.10
311	8.0606039399	**8.0560797869**	**8.0581898041**	**8.0605396501**	−4.52E−03	−2.41E−03	−6.43E−05	1/10	10/10	31 867.30
312	8.0669842552	**8.0634879963**	8.0676502918	8.0719840528	−3.50E−03	6.66E−04	5.00E−03	1/10	3/10	32 574.10
313	8.0756527378	**8.0723776565**	8.0765779098	8.0799936792	−3.28E−03	9.25E−04	4.34E−03	1/10	5/10	32 459.30
314	8.0892357495	**8.0840444646**	**8.0861982805**	**8.0881525809**	−5.19E−03	−3.04E−03	−1.08E−03	1/10	10/10	27 667.50
315	8.1013721248	**8.0833362248**	**8.0917078266**	8.1016162151	−1.80E−02	−9.66E−03	2.44E−04	1/10	9/10	35 320.70
316	8.1113304407	**8.0914221909**	**8.0974417541**	**8.1067464630**	−1.99E−02	−1.39E−02	−4.58E−03	1/10	10/10	37 634.80
317	8.1200535298	**8.0985648551**	**8.1080890701**	**8.1142830324**	−2.15E−02	−1.20E−02	−5.77E−03	1/10	10/10	34 461.70
318	8.1282408602	**8.1078206147**	**8.1160501234**	**8.1227262055**	−2.04E−02	−1.22E−02	−5.51E−03	1/10	10/10	32 015.10
319	8.1367023254	**8.1191101842**	**8.1287896608**	**8.1334730214**	−1.76E−02	−7.91E−03	−3.23E−03	1/10	10/10	28 599.30
320	8.1466752518	**8.1327700764**	**8.1382270265**	**8.1433358422**	−1.39E−02	−8.45E−03	−3.34E−03	1/10	10/10	33 270.80
321	8.1563823395	**8.1397394926**	**8.1460532738**	**8.1560281961**	−1.66E−02	−1.03E−02	−3.54E−04	1/10	10/10	35 082.60
322	8.1650140248	**8.1477840758**	**8.1557411143**	**8.1643496229**	−1.72E−02	−9.27E−03	−6.64E−04	1/10	10/10	37 939.20
323	8.1723756268	**8.1587592542**	**8.1635652644**	**8.1690624142**	−1.36E−02	−8.81E−03	−3.31E−03	1/10	10/10	26 588.60
324	8.1796327858	**8.1674085081**	**8.1718142906**	**8.1780197354**	−1.22E−02	−7.82E−03	−1.61E−03	1/10	10/10	33 026.40
325	8.1882217082	**8.1733003234**	**8.1805874525**	8.1936774151	−1.49E−02	−7.63E−03	5.46E−03	1/10	9/10	33 231.00
326	8.1975179838	**8.1817407277**	**8.1914547326**	8.2046223623	−1.58E−02	−6.06E−03	7.10E−03	1/10	8/10	29 824.20
327	8.2058236905	**8.1880247293**	**8.1979037392**	8.2103550207	−1.78E−02	−7.92E−03	4.53E−03	1/10	9/10	34 200.90
328	8.2136443647	**8.1942670899**	**8.2080642404**	8.2182671761	−1.94E−02	−5.58E−03	4.62E−03	1/10	6/10	32 810.00
329	8.2182864526	**8.2060215573**	**8.2168840449**	8.2291349103	−1.23E−02	−1.40E−03	1.08E−02	1/10	6/10	29 943.10
330	8.2298817657	**8.2157405956**	**8.2270903671**	8.2349753308	−1.41E−02	−2.79E−03	5.09E−03	1/10	6/10	38 672.90
331	8.2373988670	**8.2204413597**	**8.2322238821**	8.2434767876	−1.70E−02	−5.17E−03	6.08E−03	1/10	6/10	30 812.60
332	8.2447022853	**8.2365077203**	**8.2446694472**	8.2522554400	−8.19E−03	−3.28E−05	7.55E−03	1/10	6/10	39 201.80
333	8.2558752036	**8.2402967266**	**8.2509101559**	**8.2549683634**	−1.56E−02	−4.97E−03	−9.07E−04	1/10	10/10	36 149.70
334	8.2641607509	**8.2624688079**	8.2653822448	8.2688092008	−1.69E−03	1.22E−03	4.65E−03	1/10	2/10	41 347.50
335	8.2716242498	**8.2550643788**	**8.2687134927**	8.2758491781	−1.66E−02	−2.91E−03	4.22E−03	1/10	5/10	30 155.00
336	8.2787743871	**8.2772630110**	8.2798230222	8.2833084340	−1.51E−03	1.05E−03	4.53E−03	1/10	4/10	32 964.40
337	8.2862241400	**8.2702833068**	**8.2858113782**	8.2918633408	−1.59E−02	−4.13E−04	5.64E−03	1/10	3/10	32 532.50
338	8.2937023722	8.2939873759	8.2968423944	8.3013392350	2.85E−04	3.14E−03	7.64E−03	1/10	0/10	35 175.30
339	8.3019941224	**8.2976655457**	8.3041915428	8.3109471120	−4.33E−03	2.20E−03	8.95E−03	1/10	3/10	37 779.40
340	8.3104116379	**8.3097262711**	8.3143372656	8.3176099179	−6.85E−04	3.93E−03	7.20E−03	1/10	1/10	31 288.50
341	8.3165790385	**8.3099338147**	8.3200594131	8.3249458826	−6.65E−03	3.48E−03	8.37E−03	1/10	1/10	41 373.40
342	8.3227045402	**8.3135903730**	8.3271377294	8.3322025035	−9.11E−03	4.43E−03	9.50E−03	1/10	2/10	33 025.80
343	8.3315888726	**8.3277561232**	8.3366690796	8.3420658683	−3.83E−03	5.08E−03	1.05E−02	1/10	1/10	33 505.00
344	8.3411770056	8.3452642590	8.3479053529	8.3557960928	4.09E−03	6.73E−03	1.46E−02	1/10	0/10	37 478.70
345	8.3518472673	**8.3420522869**	8.3520364567	8.3586699178	−9.79E−03	1.89E−04	6.82E−03	1/10	5/10	34 106.30
346	8.3591153674	**8.3537759755**	8.3604435937	8.3660336803	−5.34E−03	1.33E−03	6.92E−03	1/10	3/10	31 978.60
347	8.3672718772	**8.3627048726**	8.3674485285	8.3727653983	−4.57E−03	1.77E−04	5.49E−03	1/10	7/10	37 954.30
348	8.3745515295	**8.3674634360**	8.3770975392	8.3835266621	−7.09E−03	2.55E−03	8.98E−03	1/10	3/10	33 932.60
349	8.3836313552	**8.3746367225**	8.3862908966	8.3925900868	−8.99E−03	2.66E−03	8.96E−03	1/10	2/10	37 856.70
350	8.3928242707	**8.3867647460**	**8.3926355872**	8.3989251991	−6.06E−03	−1.89E−04	6.10E−03	1/10	6/10	30 606.70
#Better		48	34	21						
#Equal		0	0	0						
#Worse		2	16	29						

Table 15
Computational results and comparisons on the instances with $351 \le n \le 400$. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (this work)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	HR	RR	$Time$ (s)
351	8.4007527076	**8.3921705143**	**8.3976610497**	8.4030706256	−8.58E−03	−3.09E−03	2.32E−03	1/10	8/10	36 231.70
352	8.4129454678	**8.4033387897**	**8.4070109767**	**8.4118404333**	−9.61E−03	−5.93E−03	−1.11E−03	1/10	10/10	32 557.90
353	8.4188352568	**8.4074380625**	**8.4173578479**	8.4244697335	−1.14E−02	−1.48E−03	5.63E−03	1/10	7/10	36 663.40
354	8.4296585287	**8.4076401001**	**8.4212974949**	8.4302859248	−2.20E−02	−8.36E−03	6.27E−04	1/10	9/10	31 278.20
355	8.4382375759	**8.4283461812**	**8.4341418972**	8.4405289554	−9.89E−03	−4.10E−03	2.29E−03	1/10	8/10	36 232.90
356	8.4440909450	**8.4267054911**	**8.4383669270**	8.4462004681	−1.74E−02	−5.72E−03	2.11E−03	1/10	7/10	35 386.90
357	8.4518644222	**8.4382769178**	**8.4499586603**	8.4554710351	−1.36E−02	−1.91E−03	3.61E−03	1/10	6/10	36 650.80
358	8.4567949768	**8.4490841743**	**8.4532273856**	8.4576399420	−7.71E−03	−3.57E−03	8.45E−04	1/10	8/10	33 469.90
359	8.4642649260	**8.4548490764**	**8.4634558943**	8.4701957415	−9.42E−03	−8.09E−04	5.93E−03	1/10	6/10	35 939.40
360	8.4721516869	**8.4651558516**	8.4726874357	8.4780898604	−7.00E−03	5.36E−04	5.94E−03	1/10	5/10	33 265.00
361	8.4805536014	**8.4645834364**	**8.4781535429**	8.4862701299	−1.60E−02	−2.40E−03	5.72E−03	1/10	6/10	37 982.70
362	8.4880679606	**8.4788009361**	8.4882055053	8.4951586532	−9.27E−03	1.38E−04	7.09E−03	1/10	3/10	36 328.70
363	8.4965064991	**8.4830627684**	**8.4963311686**	8.5058544600	−1.34E−02	−1.75E−04	9.35E−03	1/10	5/10	36 888.10
364	8.5030156544	**8.4936601180**	**8.5015802313**	8.5120190906	−9.36E−03	−1.44E−03	9.00E−03	1/10	6/10	41 510.80
365	8.5120741140	**8.4893661721**	**8.5101067518**	8.5201200433	−2.27E−02	−1.97E−03	8.05E−03	1/10	5/10	36 513.90
366	8.5207116927	**8.5142845330**	**8.5203955423**	8.5271738686	−6.43E−03	−3.16E−04	6.46E−03	1/10	6/10	38 102.30
367	8.5271184405	**8.5206107101**	8.5276469145	8.5372166183	−6.51E−03	5.28E−04	1.01E−02	1/10	5/10	33 376.70
368	8.5351429761	**8.5308028271**	8.5388016906	8.5441630173	−4.34E−03	3.66E−03	9.02E−03	1/10	1/10	29 263.30
369	8.5433733193	**8.5337041371**	8.5441638262	8.5507580458	−9.67E−03	7.91E−04	7.38E−03	1/10	3/10	33 058.30
370	8.5498955957	**8.5477264184**	8.5535890629	8.5596560704	−2.17E−03	3.69E−03	9.76E−03	1/10	1/10	38 641.60
371	8.5580474842	8.5583387631	8.5637206436	8.5684790990	2.91E−04	5.67E−03	1.04E−02	1/10	0/10	34 721.10
372	8.5646395490	**8.5618715156**	8.5688457203	8.5762215917	−2.77E−03	4.21E−03	1.16E−02	1/10	2/10	38 153.50
373	8.5718304380	**8.5712024323**	8.5784071217	8.5876835626	−6.28E−04	6.58E−03	1.59E−02	1/10	1/10	29 532.80
374	8.5786934269	**8.5774728308**	8.5851262942	8.5925454131	−1.22E−03	6.43E−03	1.39E−02	1/10	1/10	34 824.20
375	8.5855784311	**8.5820685572**	8.5927704139	8.5975982088	−3.51E−03	7.19E−03	1.20E−02	1/10	1/10	32 022.70
376	8.5940896993	8.5962563229	8.6019527602	8.6058417826	2.17E−03	7.86E−03	1.18E−02	1/10	0/10	36 583.20
377	8.6005441929	8.6027978057	8.6079290231	8.6129540210	2.25E−03	7.38E−03	1.24E−02	1/10	0/10	36 643.40
378	8.6069492023	8.6075915129	8.6167326917	8.6234197960	6.42E−04	9.78E−03	1.65E−02	1/10	0/10	36 768.70
379	8.6149479886	8.6184729253	8.6232130064	8.6284298875	3.52E−03	8.27E−03	1.35E−02	1/10	0/10	31 997.50
380	8.6213515024	**8.6167407193**	8.6297682259	8.6347852477	−4.61E−03	8.42E−03	1.34E−02	1/10	1/10	30 666.80
381	8.6290602650	8.6337617241	8.6390157316	8.6429890868	4.70E−03	9.96E−03	1.39E−02	1/10	0/10	31 068.60
382	8.6388782340	8.6408402938	8.6459287777	8.6506591264	1.96E−03	7.05E−03	1.18E−02	1/10	0/10	37 316.30
383	8.6458487389	8.6513501064	8.6540511824	8.6580287179	5.50E−03	8.20E−03	1.22E−02	1/10	0/10	33 845.70
384	8.6537986688	**8.6534665127**	8.6610714907	8.6673217416	−3.32E−04	7.27E−03	1.35E−02	1/10	1/10	34 619.30
385	8.6597468731	8.6607058843	8.6666686067	8.6710327517	9.59E−04	6.92E−03	1.13E−02	1/10	0/10	31 808.30
386	8.6688960219	8.6727731360	8.6772487889	8.6812511021	3.88E−03	8.35E−03	1.24E−02	1/10	0/10	33 264.10
387	8.6743044156	8.6831299075	8.6851125353	8.6869312952	8.83E−03	1.08E−02	1.26E−02	1/10	0/10	36 257.30
388	8.6806940940	8.6825360520	8.6900621836	8.6958625239	1.84E−03	9.37E−03	1.52E−02	1/10	0/10	32 091.90
389	8.6873874968	8.6916892658	8.6986627004	8.7030910903	4.30E−03	1.13E−02	1.57E−02	1/10	0/10	34 963.00
390	8.6956660896	8.6976501640	8.7057093833	8.7111966859	1.98E−03	1.00E−02	1.55E−02	1/10	0/10	40 716.10
391	8.7030232271	8.7036860347	8.7117841202	8.7148229518	6.63E−04	8.76E−03	1.18E−02	1/10	0/10	33 504.30
392	8.7102535414	**8.7075529636**	8.7194373948	8.7262473878	−2.70E−03	9.18E−03	1.60E−02	1/10	1/10	33 894.60
393	8.7212637605	8.7220259806	8.7267001491	8.7331599685	7.62E−04	5.44E−03	1.19E−02	1/10	0/10	38 376.70
394	8.7292656343	8.7304320881	8.7352037357	8.7394802761	1.17E−03	5.94E−03	1.02E−02	1/10	0/10	32 689.40
395	8.7348094475	**8.7307593517**	8.7414685181	8.7482729534	−4.05E−03	6.66E−03	1.35E−02	1/10	1/10	34 992.80
396	8.7405814827	**8.7383102973**	8.7473704141	8.7547355424	−2.27E−03	6.79E−03	1.42E−02	1/10	1/10	37 857.60
397	8.7466387404	8.7480616080	8.7583995060	8.7625240864	1.42E−03	1.18E−02	1.59E−02	1/10	0/10	32 398.40
398	8.7534203080	8.7595163352	8.7635387623	8.7689618626	6.10E−03	1.01E−02	1.55E−02	1/10	0/10	36 624.30
399	8.7595077675	8.7652265248	8.7728449064	8.7772272067	5.72E−03	1.33E−02	1.77E−02	1/10	0/10	33 664.50
400	8.7674798301	8.7683321754	8.7798413330	8.7832310168	8.52E−04	1.24E−02	1.58E−02	1/10	0/10	36 997.10
#Better		29	14	1						
#Equal		0	0	0						
#Worse		21	36	49						

Table 16
Computational results and comparison on the 50 moderate-scale instances ($351 \leq n \leq 400$) with the cut-off time set to 2 h. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (T_{cut} = 2 h)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	$Time$ (s)	$N_{iter.}$	$N_{expl.}$
351	8.4007527076	**8.3996465688**	8.4089380279	8.4188651497	−1.11E−03	8.19E−03	1.81E−02	5420.60	257	14 469
352	8.4129454678	**8.4091364930**	8.4160166542	8.4221447842	−3.81E−03	3.07E−03	9.20E−03	4746.00	257	14 509
353	8.4188352568	8.4194281857	8.4267811886	8.4341338982	5.93E−04	7.95E−03	1.53E−02	5169.60	249	14 078
354	8.4296585287	**8.4177166906**	8.4300961029	8.4379967245	−1.19E−02	4.38E−04	8.34E−03	5762.30	253	14 215
355	8.4382375759	**8.4319460786**	8.4395656507	8.4475168252	−6.29E−03	1.33E−03	9.28E−03	5825.00	249	13 905
356	8.4440909450	8.4443570488	8.4494354873	8.4589999677	2.66E−04	5.34E−03	1.49E−02	6020.50	246	13 875
357	8.4518644222	8.4540996284	8.4586886534	8.4642249091	2.24E−03	6.82E−03	1.24E−02	5786.30	243	13 703
358	8.4567949768	8.4580619333	8.4664580099	8.4747959653	1.27E−03	9.66E−03	1.80E−02	4896.10	247	13 838
359	8.4642649260	**8.4638789814**	8.4721116044	8.4778980494	−3.86E−04	7.85E−03	1.36E−02	5210.60	245	13 754
360	8.4721516869	**8.4714630109**	8.4812181934	8.4919534081	−6.89E−04	9.07E−03	1.98E−02	4945.00	247	13 863
361	8.4805536014	8.4822870635	8.4882002583	8.4943661562	1.73E−03	7.65E−03	1.38E−02	5969.60	243	13 551
362	8.4880679606	**8.4818924210**	8.4953124443	8.5040863347	−6.18E−03	7.24E−03	1.60E−02	4997.40	245	13 709
363	8.4965064991	**8.4945147585**	8.5054328985	8.5125202116	−1.99E−03	8.93E−03	1.60E−02	5656.20	232	13 140
364	8.5030156544	8.5042481035	8.5134538677	8.5203135564	1.23E−03	1.04E−02	1.73E−02	5733.60	235	13 285
365	8.5120741140	**8.5108442195**	8.5208710191	8.5280166171	−1.23E−03	8.80E−03	1.59E−02	4775.50	235	13 169
366	8.5207116927	**8.5185677895**	8.5280152293	8.5346713621	−2.14E−03	7.30E−03	1.40E−02	6016.90	235	13 201
367	8.5271184405	**8.5256457131**	8.5367107431	8.5464133814	−1.47E−03	9.59E−03	1.93E−02	6083.30	229	12 942
368	8.5351429761	**8.5347424089**	8.5425471503	8.5515643943	−4.01E−04	7.40E−03	1.64E−02	4854.90	232	13 035
369	8.5433733193	8.5457847244	8.5543548032	8.5650001022	2.41E−03	1.10E−02	2.16E−02	5598.20	225	12 673
370	8.5498955957	8.5505951920	8.5627566268	8.5675741529	7.00E−04	1.29E−02	1.77E−02	5807.40	223	12 501
371	8.5580474842	**8.5560064744**	8.5691342516	8.5786458511	−2.04E−03	1.11E−02	2.06E−02	5593.50	222	12 543
372	8.5646395490	8.5654625584	8.5774616895	8.5834699427	8.23E−04	1.28E−02	1.88E−02	5395.60	221	12 478
373	8.5718304380	8.5740746057	8.5851604862	8.5909710092	2.24E−03	1.33E−02	1.91E−02	3991.70	219	12 407
374	8.5786934269	8.5880998853	8.5937932470	8.6010518030	9.41E−03	1.51E−02	2.24E−02	5528.70	220	12 336
375	8.5855784311	8.5884374413	8.6005296825	8.6088406145	2.86E−03	1.50E−02	2.33E−02	5109.00	215	12 202
376	8.5940896993	8.5969864442	8.6074979711	8.6143087144	2.90E−03	1.34E−02	2.02E−02	5313.10	215	12 118
377	8.6005441929	8.6060708648	8.6172949661	8.6267921068	5.53E−03	1.68E−02	2.62E−02	5111.90	209	11 862
378	8.6069492023	8.6174409939	8.6224283422	8.6272614566	1.05E−02	1.55E−02	2.03E−02	5122.60	209	11 907
379	8.6149479886	8.6222002746	8.6283801391	8.6366447373	7.25E−03	1.34E−02	2.17E−02	5685.10	211	11 963
380	8.6213515024	8.6243660283	8.6343724505	8.6433744087	3.01E−03	1.30E−02	2.20E−02	5204.40	213	11 986
381	8.6290602650	8.6400455037	8.6450548658	8.6497695642	1.10E−02	1.60E−02	2.07E−02	4618.00	208	11 644
382	8.6388782340	8.6398611253	8.6522470431	8.6621859801	9.83E−04	1.34E−02	2.33E−02	6065.50	206	11 544
383	8.6458487389	8.6575576882	8.6618147014	8.6665670715	1.17E−02	1.60E−02	2.07E−02	5278.40	203	11 415
384	8.6537986688	8.6599260505	8.6684386588	8.6770946819	6.13E−03	1.46E−02	2.33E−02	5687.40	202	11 424
385	8.6597468731	8.6687784209	8.6774932676	8.6836135104	9.03E−03	1.77E−02	2.39E−02	5191.80	200	11 214
386	8.6688960219	8.6751271202	8.6814504824	8.6880323885	6.23E−03	1.26E−02	1.91E−02	5990.80	201	11 199
387	8.6743044156	8.6789051593	8.6882021034	8.6953261083	4.60E−03	1.39E−02	2.10E−02	6301.50	202	11 327
388	8.6806940940	8.6862794418	8.6948432569	8.7039705821	5.59E−03	1.41E−02	2.33E−02	5537.50	202	11 274
389	8.6873874968	8.6897499390	8.7032690485	8.7095021317	2.36E−03	1.59E−02	2.21E−02	4840.00	194	11 102
390	8.6956660896	8.7080886245	8.7140591275	8.7178385283	1.24E−02	1.84E−02	2.22E−02	5719.60	192	10 951
391	8.7030232271	8.7127405787	8.7206367067	8.7291083589	9.72E−03	1.76E−02	2.61E−02	4911.20	192	10 856
392	8.7102535414	8.7206822036	8.7258158187	8.7306998468	1.04E−02	1.56E−02	2.04E−02	4812.20	193	10 801
393	8.7212637605	8.7247764722	8.7332605507	8.7392287327	3.51E−03	1.20E−02	1.80E−02	5541.30	189	10 804
394	8.7292656343	8.7319163374	8.7403940151	8.7489951156	2.65E−03	1.11E−02	1.97E−02	4656.90	192	10 832
395	8.7348094475	8.7349829757	8.7482807812	8.7582269434	1.74E−04	1.35E−02	2.34E−02	4812.60	188	10 661
396	8.7405814827	8.7536690582	8.7583927485	8.7643913782	1.31E−02	1.78E−02	2.38E−02	6112.50	182	10 429
397	8.7466387404	8.7581396407	8.7648141569	8.7704855422	1.15E−02	1.82E−02	2.38E−02	5052.70	181	10 273
398	8.7534203080	8.7615899443	8.7696037619	8.7762763013	8.17E−03	1.62E−02	2.29E−02	5201.30	182	10 234
399	8.7595077675	8.7694391180	8.7773549223	8.7816216381	9.93E−03	1.78E−02	2.21E−02	4115.20	181	10 410
400	8.7674798301	8.7830646218	8.7865973848	8.7922727413	1.56E−02	1.91E−02	2.48E−02	5832.40	182	10 164
#Better		13	0	0						
#Equal		0	0	0						
#Worse		37	50	50						

Table 17
Computational results and comparison on the 50 moderate-scale instances ($351 \le n \le 400$) with the cut-off time set to 6 h. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (T_{cut} = 6 h)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	$Time$ (s)	$N_{iter.}$	$N_{expl.}$
351	8.4007527076	**8.3991945364**	8.4082621967	8.4177873151	−1.56E−03	7.51E−03	1.70E−02	15 818.10	788	45 045
352	8.4129454678	**8.4088235272**	8.4141837800	8.4175681052	−4.12E−03	1.24E−03	4.62E−03	18 742.30	779	44 846
353	8.4188352568	**8.4145497751**	8.4241184711	8.4292944428	−4.29E−03	5.28E−03	1.05E−02	17 045.00	763	44 081
354	8.4296585287	**8.4135178730**	**8.4276232579**	8.4358313518	−1.61E−02	−2.04E−03	6.17E−03	15 188.10	791	44 676
355	8.4382375759	**8.4247274204**	**8.4351661388**	8.4415790665	−1.35E−02	−3.07E−03	3.34E−03	17 449.50	779	44 173
356	8.4440909450	**8.4360304324**	8.4457185896	8.4540218641	−8.06E−03	1.63E−03	9.93E−03	15 371.60	775	43 842
357	8.4518644222	**8.4487831957**	8.4551475007	8.4642249091	−3.08E−03	3.28E−03	1.24E−02	13 981.50	762	43 178
358	8.4567949768	**8.4560292105**	8.4627253991	8.4709209982	−7.66E−04	5.93E−03	1.41E−02	17 722.40	770	44 029
359	8.4642649260	**8.4624244276**	8.4697982527	8.4771260702	−1.84E−03	5.53E−03	1.29E−02	15 565.00	760	43 202
360	8.4721516869	**8.4708687810**	8.4760670474	8.4823745284	−1.28E−03	3.92E−03	1.02E−02	17 450.70	756	43 017
361	8.4805536014	**8.4747552064**	8.4824196683	8.4888875514	−5.80E−03	1.87E−03	8.33E−03	17 837.80	762	42 867
362	8.4880679606	**8.4801282417**	8.4903326660	8.4984927951	−7.94E−03	2.26E−03	1.04E−02	13 948.50	768	43 104
363	8.4965064991	**8.4853235412**	8.5011766784	8.5080571523	−1.12E−02	4.67E−03	1.16E−02	17 978.00	733	41 791
364	8.5030156544	**8.5012413251**	8.5064852547	8.5173972698	−1.77E−03	3.47E−03	1.44E−02	19 676.70	741	42 136
365	8.5120741140	**8.5108442195**	8.5177671670	8.5280166171	−1.23E−03	5.69E−03	1.59E−02	17 792.40	739	41 571
366	8.5207116927	**8.5150859887**	8.5225660662	8.5283758674	−5.63E−03	1.85E−03	7.66E−03	17 904.00	737	41 623
367	8.5271184405	**8.5243870671**	8.5347720806	8.5430104524	−2.73E−03	7.65E−03	1.59E−02	13 565.40	718	41 005
368	8.5351429761	**8.5314678724**	8.5396932868	8.5515643943	−3.68E−03	4.55E−03	1.64E−02	19 264.80	722	40 992
369	8.5433733193	**8.5410350949**	8.5487601472	8.5531514696	−2.34E−03	5.39E−03	9.78E−03	14 348.20	712	40 317
370	8.5498955957	**8.5478904124**	8.5584242825	8.5620410323	−2.01E−03	8.53E−03	1.21E−02	17 755.10	701	39 499
371	8.5580474842	**8.5486280037**	8.5631206545	8.5749651297	−9.42E−03	5.07E−03	1.69E−02	20 444.20	718	40 497
372	8.5646395490	**8.5623342399**	8.5729894361	8.5806416763	−2.31E−03	8.35E−03	1.60E−02	16 699.00	694	39 715
373	8.5718304380	8.5740746057	8.5822865642	8.5909710092	2.24E−03	1.05E−02	1.91E−02	15 852.60	674	38 852
374	8.5786934269	8.5821114004	8.5893857349	8.5988497236	3.42E−03	1.07E−02	2.02E−02	18 772.70	691	39 743
375	8.5855784311	**8.5852362291**	8.5965106366	8.6085284157	−3.42E−04	1.09E−02	2.29E−02	17 554.80	663	38 432
376	8.5940896993	**8.5900055769**	8.6034667636	8.6143087144	−4.08E−03	9.38E−03	2.02E−02	18 553.50	675	38 639
377	8.6005441929	8.6043963983	8.6133864198	8.6196264646	3.85E−03	1.28E−02	1.91E−02	18 601.70	649	37 666
378	8.6069492023	8.6111391185	8.6200669906	8.6247919250	4.19E−03	1.31E−02	1.78E−02	18 438.20	651	37 705
379	8.6149479886	8.6158922262	8.6228288953	8.6366447373	9.44E−04	7.88E−03	2.17E−02	16 045.90	660	37 997
380	8.6213515024	8.6218049375	8.6314351969	8.6410464267	4.53E−04	1.01E−02	1.97E−02	14 818.40	660	37 770
381	8.6290602650	8.6357107409	8.6416958112	8.6475398010	6.65E−03	1.26E−02	1.85E−02	16 665.40	648	37 070
382	8.6388782340	**8.6382167273**	8.6485342310	8.6587970413	−6.62E−04	9.66E−03	1.99E−02	17 296.30	644	36 892
383	8.6458487389	8.6500970184	8.6580100275	8.6611312458	4.25E−03	1.22E−02	1.53E−02	15 570.00	634	36 524
384	8.6537986688	8.6549518407	8.6628929251	8.6690312136	1.15E−03	9.09E−03	1.52E−02	14 830.00	627	36 373
385	8.6597468731	**8.6570380866**	8.6718943041	8.6779606779	−2.71E−03	1.21E−02	1.82E−02	18 814.70	627	35 994
386	8.6688960219	**8.6675864943**	8.6759921347	8.6830437509	−1.31E−03	7.10E−03	1.41E−02	18 573.00	636	36 575
387	8.6743044156	8.6789051593	8.6847430113	8.6931441250	4.60E−03	1.04E−02	1.88E−02	17 377.10	641	36 760
388	8.6806940940	8.6841393447	8.6918887669	8.6968887975	3.45E−03	1.12E−02	1.62E−02	13 807.40	624	35 837
389	8.6873874968	**8.6850631826**	8.7013985646	8.7075304235	−2.32E−03	1.40E−02	2.01E−02	17 512.10	610	35 266
390	8.6956660896	8.7045098200	8.7092190379	8.7161501937	8.84E−03	1.36E−02	2.05E−02	15 253.10	609	34 970
391	8.7030232271	8.7092532468	8.7161960907	8.7220965516	6.23E−03	1.32E−02	1.91E−02	15 613.30	598	34 309
392	8.7102535414	8.7130898049	8.7226274494	8.7279742289	2.84E−03	1.24E−02	1.77E−02	17 616.20	599	34 260
393	8.7212637605	8.7247197734	8.7303885752	8.7382371735	3.46E−03	9.12E−03	1.70E−02	16 044.10	587	34 139
394	8.7292656343	**8.7276780434**	8.7373968177	8.7478055724	−1.59E−03	8.13E−03	1.85E−02	16 862.80	593	34 160
395	8.7348094475	**8.7322977408**	8.7445094575	8.7501360042	−2.51E−03	9.70E−03	1.53E−02	14 866.10	583	33 776
396	8.7405814827	8.7495064161	8.7534970485	8.7584036247	8.92E−03	1.29E−02	1.78E−02	18 409.80	574	33 264
397	8.7466387404	8.7574683792	8.7630073242	8.7682353490	1.08E−02	1.64E−02	2.16E−02	17 259.20	563	32 473
398	8.7534203080	8.7608664248	8.7658556080	8.7750477051	7.45E−03	1.24E−02	2.16E−02	15 602.10	562	32 532
399	8.7595077675	8.7688080563	8.7751424539	8.7804521564	9.30E−03	1.56E−02	2.09E−02	16 262.90	566	32 959
400	8.7674798301	8.7786622531	8.7830040049	8.7873800545	1.12E−02	1.55E−02	1.99E−02	17 932.50	567	32 593
#Better		30	2	0						
#Equal		0	0	0						
#Worse		20	48	50						

Table 18
Computational results and comparison on the 50 moderate-scale instances ($351 \le n \le 400$) with the cut-off time set to 12 h. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (T_{cut} = 12 h)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	$Time$ (s)	$N_{iter.}$	$N_{expl.}$
351	8.4007527076	**8.3945154809**	8.4033220390	8.4104287602	−6.24E−03	2.57E−03	9.68E−03	25 133.80	1195	89 985
352	8.4129454678	**8.4057704116**	**8.4111914970**	8.4163346618	−7.18E−03	−1.75E−03	3.39E−03	31 641.50	1102	88 425
353	8.4188352568	**8.4108773467**	8.4197911280	8.4242214461	−7.96E−03	9.56E−04	5.39E−03	30 561.00	1134	88 060
354	8.4296585287	**8.4135178730**	**8.4244738731**	8.4308783720	−1.61E−02	−5.18E−03	1.22E−03	22 511.50	1300	88 808
355	8.4382375759	**8.4239683980**	**8.4330721283**	8.4405509197	−1.43E−02	−5.17E−03	2.31E−03	22 761.50	1317	88 287
356	8.4440909450	**8.4333486287**	**8.4409271522**	8.4469698049	−1.07E−02	−3.16E−03	2.88E−03	20 232.40	1315	87 658
357	8.4518644222	**8.4453067803**	**8.4505014470**	8.4563721935	−6.56E−03	−1.36E−03	4.51E−03	28 200.00	1152	86 198
358	8.4567949768	**8.4526805450**	8.4584789172	8.4641760503	−4.11E−03	1.68E−03	7.38E−03	31 314.40	1145	86 623
359	8.4642649260	**8.4570407228**	8.4650001935	8.4750070094	−7.22E−03	7.35E−04	1.07E−02	23 339.40	1257	85 992
360	8.4721516869	**8.4656882863**	8.4725671936	8.4811094680	−6.46E−03	4.16E−04	8.96E−03	27 449.90	1230	85 485
361	8.4805536014	**8.4725636473**	**8.4796164049**	8.4886202744	−7.99E−03	−9.37E−04	8.07E−03	27 968.20	1286	85 879
362	8.4880679606	**8.4801282417**	8.4881362654	8.4939958224	−7.94E−03	6.83E−05	5.93E−03	22 799.10	1255	85 816
363	8.4965064991	**8.4850599235**	**8.4962447326**	8.5037728302	−1.14E−02	−2.62E−04	7.27E−03	29 371.80	1128	84 012
364	8.5030156544	**8.4943206659**	8.5031436529	8.5075028096	−8.69E−03	1.28E−04	4.49E−03	28 683.80	1162	83 940
365	8.5120741140	**8.5043586166**	**8.5109635746**	8.5179793249	−7.72E−03	−1.11E−03	5.91E−03	27 903.90	1219	83 226
366	8.5207116927	**8.5112300788**	**8.5194681190**	8.5257286331	−9.48E−03	−1.24E−03	5.02E−03	24 705.80	1272	83 208
367	8.5271184405	**8.5205601987**	8.5299900792	8.5354967127	−6.56E−03	2.87E−03	8.38E−03	28 190.50	1144	82 306
368	8.5351429761	**8.5302499922**	8.5360377652	8.5439635962	−4.89E−03	8.95E−04	8.82E−03	29 513.00	1190	81 727
369	8.5433733193	**8.5351839789**	**8.5432561810**	8.5529807964	−8.19E−03	−1.17E−04	9.61E−03	23 836.30	1217	80 914
370	8.5498955957	**8.5404315001**	8.5534802077	8.5590175752	−9.46E−03	3.58E−03	9.12E−03	29 829.10	1146	79 269
371	8.5580474842	**8.5482896386**	**8.5577014998**	8.5715860798	−9.76E−03	−3.46E−04	1.35E−02	24 871.10	1294	81 579
372	8.5646395490	**8.5602400127**	8.5690422376	8.5757725393	−4.40E−03	4.40E−03	1.11E−02	21 003.90	1237	79 042
373	8.5718304380	**8.5660176574**	8.5785685355	8.5829351326	−5.81E−03	6.74E−03	1.11E−02	28 944.30	1114	77 671
374	8.5786934269	8.5788247276	8.5841436956	8.5887173255	1.31E−04	5.45E−03	1.00E−02	34 259.50	1020	78 896
375	8.5855784311	**8.5851714255**	8.5915310323	8.6022967405	−4.07E−04	5.95E−03	1.67E−02	28 344.10	1096	77 529
376	8.5940896993	**8.5865057727**	8.5947600957	8.6022244321	−7.58E−03	6.70E−04	8.13E−03	21 568.30	1277	78 141
377	8.6005441929	8.6030157264	8.6079364105	8.6135732024	2.47E−03	7.39E−03	1.30E−02	32 566.00	1071	75 159
378	8.6069492023	**8.6068106608**	8.6124123092	8.6174820302	−1.39E−04	5.46E−03	1.05E−02	22 954.80	1232	76 060
379	8.6149479886	**8.6087193733**	8.6166135017	8.6247358849	−6.23E−03	1.67E−03	9.79E−03	22 093.70	1313	76 897
380	8.6213515024	**8.6191445370**	8.6282964332	8.6387865325	−2.21E−03	6.94E−03	1.74E−02	30 907.50	1217	75 354
381	8.6290602650	8.6297937917	8.6354931672	8.6426255486	7.34E−04	6.43E−03	1.36E−02	24 620.00	1178	74 749
382	8.6388782340	**8.6310089777**	8.6394534786	8.6505219511	−7.87E−03	5.75E−04	1.16E−02	26 789.80	1155	74 845
383	8.6458487389	**8.6414087792**	8.6514004139	8.6584529092	−4.44E−03	5.55E−03	1.26E−02	32 436.60	1103	73 893
384	8.6537986688	**8.6473244636**	8.6579840879	8.6639093236	−6.47E−03	4.19E−03	1.01E−02	24 613.40	1098	72 785
385	8.6597468731	**8.6558895018**	8.6646350225	8.6727830561	−3.86E−03	4.89E−03	1.30E−02	23 555.00	1114	72 697
386	8.6688960219	**8.6600870637**	8.6689985042	8.6788385944	−8.81E−03	1.02E−04	9.94E−03	24 154.10	1271	74 391
387	8.6743044156	**8.6682364929**	8.6785544947	8.6842976367	−6.07E−03	4.25E−03	9.99E−03	29 485.40	1186	73 605
388	8.6806940940	**8.6772501701**	8.6875302337	8.6932193466	−3.44E−03	6.84E−03	1.25E−02	23 836.90	1103	71 653
389	8.6873874968	**8.6799482207**	8.6933645462	8.7048546709	−7.44E−03	5.98E−03	1.75E−02	23 616.60	1185	70 961
390	8.6956660896	**8.6934395921**	8.7034152047	8.7094835027	−2.23E−03	7.75E−03	1.38E−02	31 684.30	1086	70 334
391	8.7030232271	**8.7021824180**	8.7121789267	8.7188924306	−8.41E−04	9.16E−03	1.59E−02	30 712.50	1026	69 032
392	8.7102535414	**8.7076108328**	8.7166486806	8.7233245660	−2.64E−03	6.40E−03	1.31E−02	27 305.10	1107	69 703
393	8.7212637605	8.7220440675	8.7249304886	8.7315461029	7.80E−04	3.67E−03	1.03E−02	30 220.20	1051	69 096
394	8.7292656343	**8.7253409203**	8.7308014081	8.7400736869	−3.92E−03	1.54E−03	1.08E−02	28 299.10	1131	68 900
395	8.7348094475	**8.7314383339**	8.7398771171	8.7501360042	−3.37E−03	5.07E−03	1.53E−02	31 849.90	1069	68 161
396	8.7405814827	**8.7363998078**	8.7464832447	8.7552114708	−4.18E−03	5.90E−03	1.46E−02	28 265.10	1119	67 530
397	8.7466387404	8.7543810172	8.7565925327	8.7594770058	7.74E−03	9.95E−03	1.28E−02	31 321.20	985	65 329
398	8.7534203080	**8.7490742205**	8.7616814910	8.7732182535	−4.35E−03	8.26E−03	1.98E−02	34 931.50	979	65 741
399	8.7595077675	8.7609920714	8.7672910556	8.7772849831	1.48E−03	7.78E−03	1.78E−02	25 891.90	1109	66 855
400	8.7674798301	**8.7581904773**	8.7735623486	8.7815724823	−9.29E−03	6.08E−03	1.41E−02	24 857.60	1110	66 175
#Better		44	11	0						
#Equal		0	0	0						
#Worse		6	39	50						

Table 19
Computational results and comparison on the 50 moderate-scale instances ($351 \leq n \leq 400$) with the cut-off time set to 24 h. In terms of R_{best}, R_{avg}, and R_{worst}, the better results appear in bold compared with the best-known results R^*, and the equal results appear in underline.

n	R^*	SED (T_{cut} = 24 h)								
		R_{best}	R_{avg}	R_{worst}	Δ_{best}	Δ_{avg}	Δ_{worst}	$Time$ (s)	$N_{iter.}$	$N_{expl.}$
351	8.4007527076	**8.3940564303**	**8.4002646386**	8.4069131206	−6.70E−03	−4.88E−04	6.16E−03	41 465.50	1966	180 432
352	8.4129454678	**8.4004580829**	**8.4065174189**	**8.4113337846**	−1.25E−02	−6.43E−03	−1.61E−03	57 853.50	1517	177 302
353	8.4188352568	**8.4090274580**	**8.4150088693**	8.4218098295	−9.81E−03	−3.83E−03	2.97E−03	58 836.10	1701	177 277
354	8.4296585287	**8.4135178730**	**8.4228254645**	**8.4274527899**	−1.61E−02	−6.83E−03	−2.21E−03	53 032.80	2153	178 112
355	8.4382375759	**8.4239683980**	**8.4321019602**	**8.4367699937**	−1.43E−02	−6.14E−03	−1.47E−03	40 548.20	2278	177 276
356	8.4440909450	**8.4328444167**	**8.4391004138**	8.4457134020	−1.12E−02	−4.99E−03	1.62E−03	53 142.00	2049	175 726
357	8.4518644222	**8.4438387381**	**8.4476328592**	8.4529391842	−8.03E−03	−4.23E−03	1.07E−03	61 289.10	1656	173 110
358	8.4567949768	**8.4424537254**	**8.4544466006**	8.4632827116	−1.43E−02	−2.35E−03	6.49E−03	57 990.80	1648	172 572
359	8.4642649260	**8.4570407228**	**8.4633137795**	8.4687409902	−7.22E−03	−9.51E−04	4.48E−03	25 941.40	2133	172 744
360	8.4721516869	**8.4563755742**	**8.4687129484**	8.4764138100	−1.58E−02	−3.44E−03	4.26E−03	57 224.00	2023	172 490
361	8.4805536014	**8.4725636473**	**8.4770419823**	8.4842773671	−7.99E−03	−3.51E−03	3.72E−03	44 265.20	2195	172 757
362	8.4880679606	**8.4801282417**	**8.4858675172**	8.4891188896	−7.94E−03	−2.20E−03	1.05E−03	45 449.00	1852	170 927
363	8.4965064991	**8.4850599235**	**8.4922029787**	**8.4959449764**	−1.14E−02	−4.30E−03	−5.62E−04	56 870.40	1693	169 379
364	8.5030156544	**8.4914009159**	**8.4981184642**	**8.5021474577**	−1.16E−02	−4.90E−03	−8.68E−04	48 741.30	1866	168 950
365	8.5120741140	**8.5007967907**	**8.5086667578**	8.5146578856	−1.13E−02	−3.41E−03	2.58E−03	56 277.30	1938	167 331
366	8.5207116927	**8.5091859536**	**8.5168807116**	8.5249038470	−1.15E−02	−3.83E−03	4.19E−03	50 078.90	2021	166 246
367	8.5271184405	**8.5175167006**	**8.5241053481**	8.5313550403	−9.60E−03	−3.01E−03	4.24E−03	58 882.70	1656	165 861
368	8.5351429761	**8.5270427988**	**8.5321017483**	8.5372534447	−8.10E−03	−3.04E−03	2.11E−03	55 037.80	1657	164 666
369	8.5433733193	**8.5313264244**	**8.5394115044**	8.5435105281	−1.20E−02	−3.96E−03	1.37E−04	51 487.00	1984	163 552
370	8.5498955957	**8.5401310897**	8.5499638831	8.5579914842	−9.76E−03	6.83E−05	8.10E−03	55 528.60	1686	159 021
371	8.5580474842	**8.5342233936**	**8.5529058144**	8.5652445509	−2.38E−02	−5.14E−03	7.20E−03	58 906.80	2066	163 430
372	8.5646395490	**8.5602400127**	8.5661145997	8.5715021650	−4.40E−03	1.48E−03	6.86E−03	67 076.40	1831	157 251
373	8.5718304380	**8.5660176574**	8.5743471866	8.5795162049	−5.81E−03	2.52E−03	7.69E−03	64 119.90	1698	155 471
374	8.5786934269	**8.5715371341**	8.5804095599	8.5854070643	−7.16E−03	1.72E−03	6.71E−03	61 445.50	1308	157 400
375	8.5855784311	**8.5751239322**	**8.5854073230**	8.5927472079	−1.05E−02	−1.71E−04	7.17E−03	60 927.50	1522	156 266
376	8.5940896993	**8.5865057727**	**8.5922076223**	8.5964129168	−7.58E−03	−1.88E−03	2.32E−03	43 923.40	2175	157 091
377	8.6005441929	**8.5952650116**	8.6032956069	8.6086551517	−5.28E−03	2.75E−03	8.11E−03	64 893.40	1479	151 145
378	8.6069492023	**8.5993943522**	8.6087450134	8.6165652558	−7.55E−03	1.80E−03	9.62E−03	63 245.70	1939	153 060
379	8.6149479886	**8.6086773553**	8.6158310374	8.6194534781	−6.27E−03	8.83E−04	4.51E−03	28 286.00	2371	155 206
380	8.6213515024	**8.6179291061**	8.6236129402	8.6301856794	−3.42E−03	2.26E−03	8.83E−03	65 912.90	1784	151 083
381	8.6290602650	**8.6260578765**	8.6323674734	8.6402360243	−3.00E−03	3.31E−03	1.12E−02	50 305.00	1885	150 232
382	8.6388782340	**8.6309931654**	**8.6372185282**	8.6438894141	−7.89E−03	−1.66E−03	5.01E−03	39 253.30	1957	150 819
383	8.6458487389	**8.6322034408**	8.6459368415	8.6558119693	−1.36E−02	8.81E−05	9.96E−03	55 095.90	1689	150 081
384	8.6537986688	**8.6473244403**	8.6540807260	8.6591177545	−6.47E−03	2.82E−04	5.32E−03	60 463.60	1616	146 102
385	8.6597468731	**8.6558895018**	8.6600540750	8.6644372961	−3.86E−03	3.07E−04	4.69E−03	51 560.80	1806	146 331
386	8.6688960219	**8.6600870637**	**8.6664628998**	8.6717011915	−8.81E−03	−2.43E−03	2.81E−03	43 908.00	2298	149 556
387	8.6743044156	**8.6634985454**	8.6753050853	8.6803227277	−1.08E−02	1.00E−03	6.02E−03	69 240.40	1761	146 653
388	8.6806940940	**8.6738491202**	8.6847083792	8.6887421630	−6.84E−03	4.01E−03	8.05E−03	55 353.70	1754	143 658
389	8.6873874968	**8.6799482207**	8.6908102168	8.7014054027	−7.44E−03	3.42E−03	1.40E−02	36 364.20	2133	142 263
390	8.6956660896	**8.6871876704**	8.6992331692	8.7059651088	−8.48E−03	3.57E−03	1.03E−02	53 929.30	1529	141 341
391	8.7030232271	**8.6970646329**	8.7070926997	8.7178302803	−5.96E−03	4.07E−03	1.48E−02	64 640.60	1504	139 240
392	8.7102535414	**8.7076108328**	8.7144592469	8.7190301952	−2.64E−03	4.21E−03	8.78E−03	49 353.30	1795	140 500
393	8.7212637605	**8.7172997574**	**8.7207368306**	8.7275748219	−3.96E−03	−5.27E−04	6.31E−03	58 944.70	1587	140 398
394	8.7292656343	**8.7231479935**	**8.7274311810**	8.7326651443	−6.12E−03	−1.83E−03	3.40E−03	53 913.40	1767	138 001
395	8.7348094475	**8.7262329954**	**8.7343797125**	8.7392201159	−8.58E−03	−4.30E−04	4.41E−03	60 446.40	1607	136 938
396	8.7405814827	**8.7363998078**	8.7428288069	8.7526965532	−4.18E−03	2.25E−03	1.21E−02	64 379.70	1837	135 246
397	8.7466387404	**8.7356896248**	8.7489208880	8.7558461943	−1.09E−02	2.28E−03	9.21E−03	62 735.80	1226	132 283
398	8.7534203080	**8.7455179548**	8.7561824822	8.7618382165	−7.90E−03	2.76E−03	8.42E−03	65 206.80	1371	132 648
399	8.7595077675	**8.7548261563**	8.7616066080	8.7688445131	−4.68E−03	2.10E−03	9.34E−03	56 215.10	1704	134 467
400	8.7674798301	**8.7581904773**	8.7700248470	8.7779811658	−9.29E−03	2.55E−03	1.05E−02	52 548.20	1769	133 018
#Better		50	27	5						
#Equal		0	0	0						
#Worse		0	23	45						

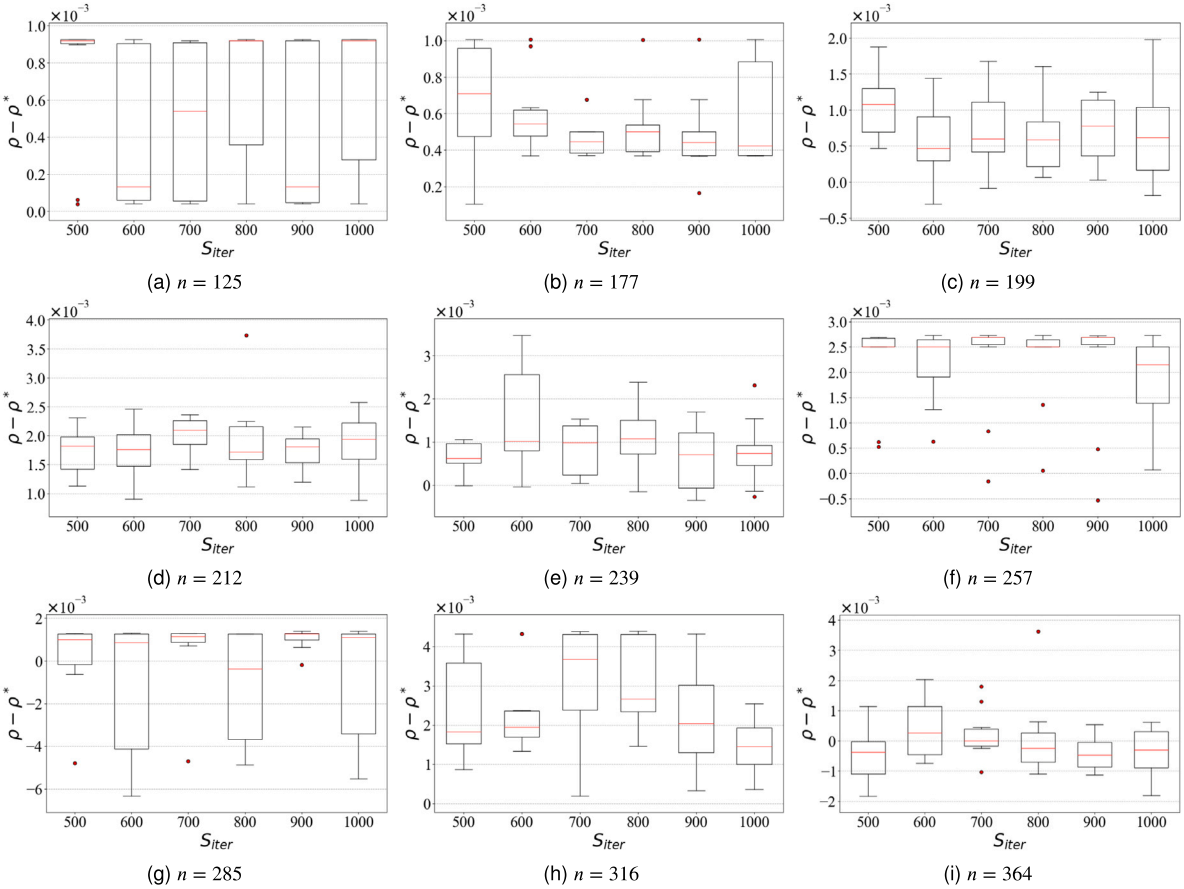


Fig. 9. Influence of parameter S_{iter} on our algorithm's performance for the nine representative instances.

Appendix B. Sensitivity analysis of parameters

Our algorithm employs three tuning parameters S_{iter}, c, and θ, which are discussed in Section 5.4. We further provide a sensitivity analysis of these parameters and discuss their influences on the algorithm.

B.1. Sensitivity analysis of parameter S_{iter}

We varied S_{iter} in the range of $\{500, 600, 700, 800, 900, 1000\}$ and examined our algorithm with these S_{iter} values on nine tested instances selected from $100 \leq n \leq 400$. The algorithm performed 10 runs with each setting value to solve each instance. The experimental results are summarized in Fig. 9 using the popular box and whisker plots, where the X-axis indicates the values of parameter S_{iter} and the Y-axis indicates the difference between the packing density ρ obtained by our algorithm and the packing density ρ^* of the best-known record from Packomania. Note that the packing density of a feasible configuration is formulated in Eq. (6).

Fig. 9 shows that the algorithm is statistically sensitive to the setting of parameter S_{iter}. However, the optimal setting value varies on different instances. Therefore, it is difficult to determine the best setting value of parameter S_{iter} for all instances. We choose the setting with $S_{iter} = 700$ as default according to the discussion in Section 5.4.

B.2. Sensitivity analysis of parameter c

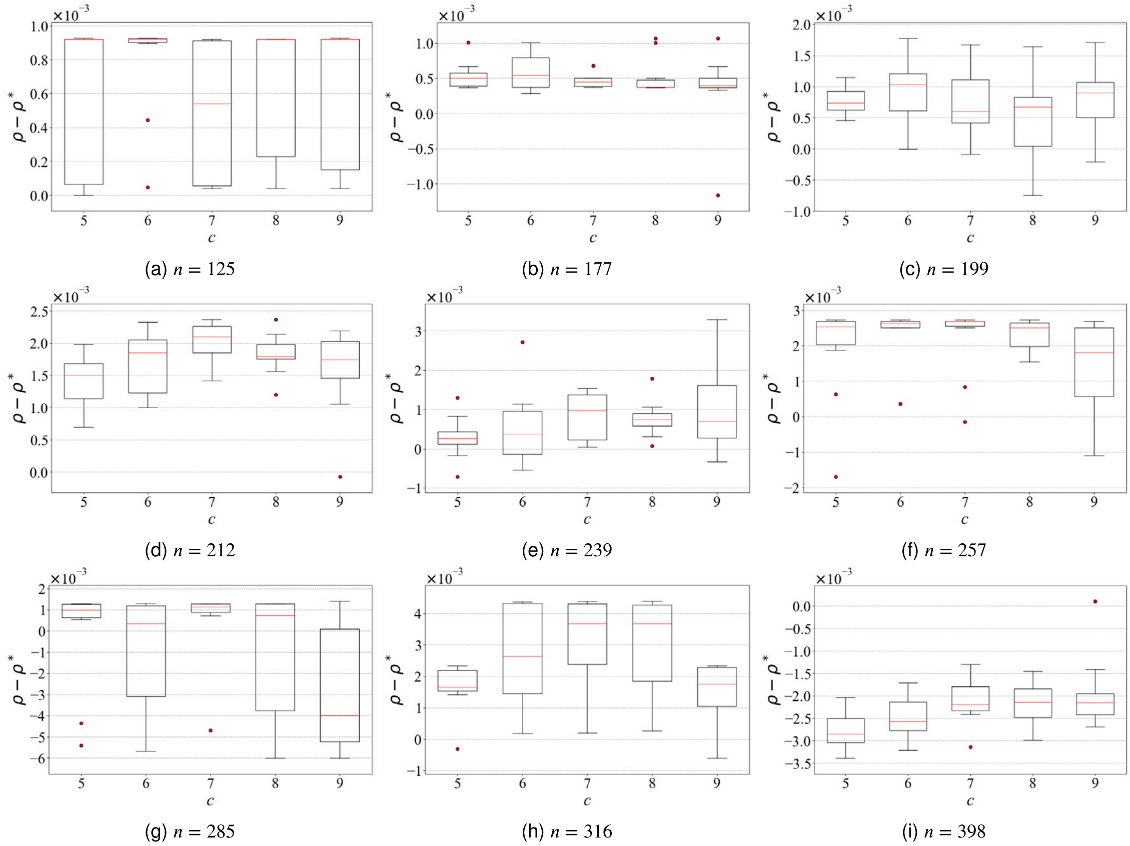
Similar to the previous experiment of parameter S_{iter}, we varied c in the range of $\{5, 6, 7, 8, 9\}$ and examined our algorithm with these c values on nine tested instances selected from $100 \leq n \leq 400$. The algorithm performed 10 runs with each setting value to solve each instance. The experimental results are summarized in Fig. 10 using the popular box and whisker plots, where the X-axis indicates the values of parameter c and the Y-axis indicates the difference between the packing density ρ obtained by our algorithm and the packing density ρ^* of the best-known record from Packomania.

Fig. 10 shows that the algorithm is statistically sensitive to the setting of parameter c. However, similar to the parameter S_{iter}, the optimal setting value varies on different instances. Therefore, it is difficult to determine the best setting value of parameter c for all instances. We choose the setting with $c = 7$ as default according to the discussion in Section 5.4.

B.3. Sensitivity analysis of parameter θ

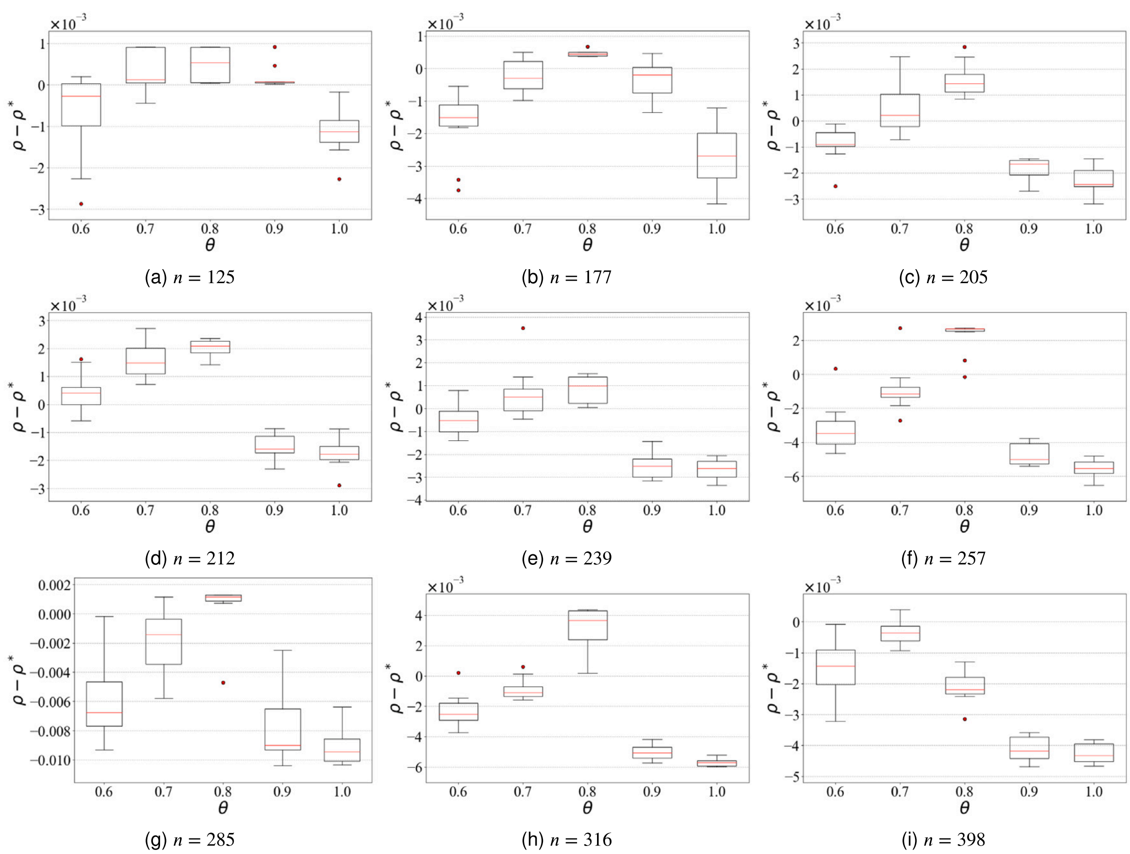
Similar to the previous experiments, we varied θ in the range of $\{0.6, 0.7, 0.8, 0.9, 1.0\}$ and examined our algorithm with these θ values on nine tested instances selected from $100 \leq n \leq 400$. The algorithm performed 10 runs with each setting value to solve each instance. The experimental results are summarized in Fig. 11 using the popular box and whisker plots, where the X-axis indicates the values of parameter c and the Y-axis indicates the difference between the packing density ρ obtained by our algorithm and the packing density ρ^* of the best-known record from Packomania.

Fig. 11 shows that the algorithm is statistically sensitive to the setting of parameter θ. In particular, the setting with $\theta = 0.8$ performs better on all the tested instances except $n = 398$, which gains a higher average packing density than other setting values on eight out of nine



(a) $n = 125$ (b) $n = 177$ (c) $n = 199$
(d) $n = 212$ (e) $n = 239$ (f) $n = 257$
(g) $n = 285$ (h) $n = 316$ (i) $n = 398$

Fig. 10. Influence of parameter c on our algorithm's performance for the nine representative instances.



(a) $n = 125$ (b) $n = 177$ (c) $n = 205$
(d) $n = 212$ (e) $n = 239$ (f) $n = 257$
(g) $n = 285$ (h) $n = 316$ (i) $n = 398$

Fig. 11. Influence of parameter θ on our algorithm's performance for the nine representative instances.

tested instances. As a result, choosing the setting with $\theta = 0.8$ as the default is the most appropriate for our algorithm.

References

Akeb, H., 2016. A two-stage look-ahead heuristic for packing spheres into a three-dimensional bin of minimum length. In: Recent Advances in Computational Optimization: Results of the Workshop on Computational Optimization WCO 2014. Springer, pp. 127–144.

Aste, T., Saadatfar, M., Senden, T., 2005. Geometrical structure of disordered sphere packings. Phys. Rev. E 71 (6), 061302.

Bezdek, K., 2012. Contact numbers for congruent sphere packings in Euclidean 3-space. Discrete Comput. Geom. 48 (2), 298–309.

Birgin, E.G., Sobral, F.N.C., 2008. Minimizing the object dimensions in circle and sphere packing problems. Comput. Oper. Res. 35 (7), 2357–2375.

Brubach, B., 2015. Improved bound for online square-into-square packing. In: Approximation and Online Algorithms: 12th International Workshop, WAOA 2014, WrocłAw, Poland, September 11-12, 2014, Revised Selected Papers. Springer, pp. 47–58.

Chaaban, A., Morvan, J.-M., Alouini, M.-S., 2016. Free-space optical communications: Capacity bounds, approximations, and a new sphere-packing perspective. IEEE Trans. Commun. 64 (3), 1176–1191.

Chen, Z., Zhao, Y., 2022. A quasi-physical method for random packing of spherical particles. Powder Technol. 412, 118002.

Cheng, H.-C., Hsieh, M.-H., Tomamichel, M., 2019. Quantum sphere-packing bounds with polynomial prefactors. IEEE Trans. Inform. Theory 65 (5), 2872–2898.

Clarke, A.S., Jónsson, H., 1993. Structural changes accompanying densification of random hard-sphere packings. Phys. Rev. E 47 (6), 3975.

Cohn, H., 2017. A conceptual breakthrough in sphere packing. Notices Amer. Math. Soc. 64 (2), 102–115.

Cohn, H., Elkies, N., 2003. New upper bounds on sphere packings I. Ann. of Math. 689–714.

Cohn, H., Kumar, A., Miller, S., Radchenko, D., Viazovska, M., 2017. The sphere packing problem in dimension 24. Ann. of Math. 185 (3), 1017–1033.

Cohn, H., Zhao, Y., 2014. Sphere packing bounds via spherical codes. Duke Math. J. 163 (10), 1965–2002.

Demaine, E.D., et al., 2016. Circle packing for origami design is hard. In: Origami 5. AK Peters/CRC Press, pp. 623–640.

Epstein, L., 2019. A lower bound for online rectangle packing. J. Comb. Optim. 38, 846–866.

Fazeli, A., Vardy, A., Yaakobi, E., 2014. Generalized sphere packing bound: Applications. In: 2014 IEEE International Symposium on Information Theory. IEEE, pp. 1261–1265.

Fazeli, A., Vardy, A., Yaakobi, E., 2015. Generalized sphere packing bound. IEEE Trans. Inform. Theory 61 (5), 2313–2334.

Fekete, S.P., von Höveling, S., Scheffer, C., 2019. Online circle packing. In: Algorithms and Data Structures: 16th International Symposium, WADS 2019, Edmonton, AB, Canada, August 5–7, 2019, Proceedings 16. Springer, pp. 366–379.

Fowler, R.J., Paterson, M.S., Tanimoto, S.L., 1981. Optimal packing and covering in the plane are NP-complete. Inform. Process. Lett. 12 (3), 133–137.

Han, K., Feng, Y., Owen, D., 2005. Sphere packing with a geometric based compression algorithm. Powder Technol. 155 (1), 33–41.

Hartman, T., Mazáč, D., Rastelli, L., 2019. Sphere packing and quantum gravity. J. High Energy Phys. 2019 (12).

He, K., Tole, K., Ni, F., Yuan, Y., Liao, L., 2021. Adaptive large neighborhood search for solving the circle bin packing problem. Comput. Oper. Res. 127, 105140.

He, K., Ye, H., Wang, Z., Liu, J., 2018. An efficient quasi-physical quasi-human algorithm for packing equal circles in a circular container. Comput. Oper. Res. 92, 26–36.

Hifi, M., Lazure, D., Yousef, L., 2017. Solving packing identical spheres into a smallest sphere with a particle swarm optimization. In: 2017 4th International Conference on Control, Decision and Information Technologies. CoDIT, IEEE, pp. 0892–0897.

Hifi, M., Mohamed-Youssouf, A., Yousef, L., 2023. A threshold search-based population algorithm for the sphere packing problem. Knowl.-Based Syst. 261, 110177.

Hifi, M., Yousef, L., 2014. Width beam and hill-climbing strategies for the three-dimensional sphere packing problem. In: 2014 Federated Conference on Computer Science and Information Systems. IEEE, pp. 421–428.

Hifi, M., Yousef, L., 2015. A dichotomous search-based heuristic for the three-dimensional sphere packing problem. Cogent Eng. 2 (1), 994257.

Hifi, M., Yousef, L., 2016. Handling lower bound and hill-climbing strategies for sphere packing problems. In: Recent Advances in Computational Optimization: Results of the Workshop on Computational Optimization. WCO 2014, Springer, pp. 145–164.

Hifi, M., Yousef, L., 2018a. A global dichotomous search-based heuristic for the three-dimensional sphere packing problem. Int. J. Oper. Res. 33 (2), 139–160.

Hifi, M., Yousef, L., 2018b. A hybrid algorithm for packing identical spheres into a container. Expert Syst. Appl. 96, 249–260.

Hifi, M., Yousef, L., 2019. A local search-based method for sphere packing problems. European J. Oper. Res. 274 (2), 482–500.

Hifi, M., Youssouf, A.M., Yousef, L., 2022. A population-based algorithm for the sphere packing problem. In: 2022 8th International Conference on Control, Decision and Information Technologies, Vol. 1. CoDIT, IEEE, pp. 1455–1460.

Hokama, P., Miyazawa, F.K., Schouery, R.C., 2016. A bounded space algorithm for online circle packing. Inform. Process. Lett. 116 (5), 337–342.

Huang, W., Xu, R., 1999. Two personification strategies for solving circles packing problem. Sci. China Ser. E 42 (6), 595–602.

Huang, W., Yu, L., 2011. A quasi physical method for the equal sphere packing problem. In: 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications. IEEE, pp. 1684–1690.

Kazakov, A., Lempert, A., Ta, T.T., 2018. The sphere packing problem into bounded containers in three-dimension non-Euclidean space. IFAC-PapersOnLine 51 (32), 782–787.

Klumov, B.A., Jin, Y., Makse, H.A., 2014. Structural properties of dense hard sphere packings. J. Phys. Chem. B 118 (36), 10761–10766.

Klumov, B.A., Khrapak, S., Morfill, G., 2011. Structural properties of dense hard sphere packings. Phys. Rev. B 83 (18), 184105.

Labra, C., Onate, E., 2009. High-density sphere packing for discrete element method simulations. Commun. Numer. Methods. Eng. 25 (7), 837–849.

Lai, X., Hao, J.-K., Xiao, R., Glover, F., 2023. Perturbation-based thresholding search for packing equal circles and spheres. INFORMS J. Comput..

Lai, X., Hao, J.-K., Yue, D., Lü, Z., Fu, Z.-H., 2022. Iterated dynamic thresholding search for packing equal circles into a circular container. European J. Oper. Res. 299 (1), 137–153.

Leech, J., Sloane, N., 1971. Sphere packings and error-correcting codes. Canad. J. Math. 23 (4), 718–745.

Lintzmayer, C.N., Miyazawa, F.K., Xavier, E.C., 2019. Online circle and sphere packing. Theoret. Comput. Sci. 776, 75–94.

Liu, W., Jin, Y., Chen, S., Makse, H.A., Li, S., 2017. Equation of state for random sphere packings with arbitrary adhesion and friction. Soft Matter 13 (2), 421–427.

Liu, D.C., Nocedal, J., 1989. On the limited memory BFGS method for large scale optimization. Math. Program. 45, 503–528.

Liu, J., Yao, Y., Zheng, Y., Geng, H., Zhou, G., 2009. An effective hybrid algorithm for the circles and spheres packing problems. In: Combinatorial Optimization and Applications: Third International Conference, COCOA 2009, Huangshan, China, June 10-12, 2009. Proceedings 3. Springer, pp. 135–144.

M'Hallah, R., Alkandari, A., 2012. Packing unit spheres into a cube using VNS. Electron. Notes Discrete Math. 39, 201–208.

M'Hallah, R., Alkandari, A., Mladenovic, N., 2013. Packing unit spheres into the smallest sphere using VNS and NLP. Comput. Oper. Res. 40 (2), 603–615.

Mueller, G.E., 2005. Numerically packing spheres in cylinders. Powder Technol. 159 (2), 105–110.

O'Toole, P.I., Hudson, T.S., 2011. New high-density packings of similarly sized binary spheres. J. Phys. Chem. C 115 (39), 19037–19040.

Shi, Y., Zhang, Y., 2008. Simulation of random packing of spherical particles with different size distributions. Appl. Phys. A 92, 621–626.

Silbert, L.E., Ertaş, D., Grest, G.S., Halsey, T.C., Levine, D., 2002. Geometry of frictionless and frictional sphere packings. Phys. Rev. E 65 (3), 031304.

Soontrapa, K., Chen, Y., 2013. Mono-sized sphere packing algorithm development using optimized Monte Carlo technique. Adv. Powder Technol. 24 (6), 955–961.

Specht, E., 2023. Packomania website. http://www.packomania.com.

Stoyan, Y., Yaskov, G., 2012. Packing congruent hyperspheres into a hypersphere. J. Global Optim. 52, 855–868.

Stoyan, Y., Yaskov, G., 2013. Packing congruent spheres into a multi-connected polyhedral domain. Int. Trans. Oper. Res. 20 (1), 79–99.

Stoyan, Y., Yaskov, G., Romanova, T., Litvinchev, I., Yakovlev, S., Cantú, J.M.V., 2020. Optimized packing multidimensional hyperspheres: A unified approach. Math. Biosci. Eng. 17 (6), 6601–6630.

Valembois, A., Fossorier, M.P., 2004. Sphere-packing bounds revisited for moderate block lengths. IEEE Trans. Inform. Theory 50 (12), 2998–3014.

Viazovska, M.S., 2017. The sphere packing problem in dimension 8. Ann. of Math. 991–1015.

Wang, J., 1999. Packing of unequal spheres and automated radiosurgical treatment planning. J. Comb. Optim. 3, 453–463.

Wouterse, A., Philipse, A., 2006. Geometrical cluster ensemble analysis of random sphere packings. J. Chem. Phys. 125 (19), 194709.

Ye, D., Han, X., Zhang, G., 2009. A note on online strip packing. J. Comb. Optim. 17 (4), 417–423.

Ye, D., Han, X., Zhang, G., 2011. Online multiple-strip packing. Theoret. Comput. Sci. 412 (3), 233–239.

Zeng, Z.Z., Huang, W.Q., Xu, R.C., Fu, Z.H., 2012. An algorithm to packing unequal spheres in a larger sphere. Adv. Mater. Res. 546, 1464–1469.