algorithms

Article

Nonlinear Optimization and Adaptive Heuristics for Solving Irregular Object Packing Problems

János D. Pintér [1,*], Ignacio Castillo [2] and Frank J. Kampas [3]

[1] Department of Management Science and Information Systems, Rutgers University, Piscataway, NJ 08854, USA
[2] Lazaridis School of Business and Economics, Wilfrid Laurier University, Waterloo, ON N2L 3C7, Canada; icastillo@wlu.ca
[3] Physicist at Large Consulting LLC, Bryn Mawr, PA 19010, USA; frank@physicistatlarge.com
* Correspondence: jpinter@business.rutgers.edu

Abstract: We review and present several challenging model classes arising in the context of finding optimized object packings (OP). Except for the smallest and/or simplest general OP model instances, it is not possible to find their exact (closed-form) solution. Most OP problem instances become increasingly difficult to handle even numerically, as the number of packed objects increases. Specifically, here we consider classes of general OP problems that can be formulated in the framework of nonlinear optimization. Research experience demonstrates that—in addition to utilizing general-purpose nonlinear optimization solver engines—the insightful exploitation of problem-specific heuristics can improve the quality of numerical solutions. We discuss scalable OP problem classes aimed at packing general circles, spheres, ellipses, and ovals, with numerical (conjectured) solutions of non-trivial model instances. In addition to their practical relevance, these models and their various extensions can also serve as constrained global optimization test challenges.

Keywords: optimized object packings; general (irregular) packings; scalable models of packing circles, spheres, ellipses, and ovals; model implementations; nonlinear optimization; heuristics; illustrative numerical results

check for updates

Citation: Pintér, J.D.; Castillo, I.; Kampas, F.J. Nonlinear Optimization and Adaptive Heuristics for Solving Irregular Object Packing Problems. *Algorithms* **2024**, *17*, 480. https://doi.org/10.3390/a17110480

Academic Editor: Roberto Montemanni

Received: 18 July 2024
Revised: 28 September 2024
Accepted: 16 October 2024
Published: 25 October 2024

1. Introduction, Technical Challenges, and Contributions

1.1. Introduction

Given a finite collection of objects, a typical packing goal is to place these objects in a non-overlapping configuration into an optimized container(s). This concise generic problem statement covers a vast range of packing problems of theoretical and practical significance. Finding optimized object packings is among the significant applications of Operations Research and Management Science.

Object-packing problems, questions, and puzzles have been of interest for a long time. Aste and Weaire [1] explore the history of packing various objects and structures in the context of mathematics, physics, chemistry, and biology. To appreciate some packing challenges, visit, e.g., the websites of Friedman [2] and Specht [3], or Wikipedia's topical page [4], with links to a selection of further websites.

Many of the packing problems presented by these websites deal with packing *identical* objects—such as triangles, squares, circles, and spheres—into two- or three-dimensional optimized containers such as minimal-size polygons, circles, semi-circles, spheres, and boxes. In some alternative model formulations, given a certain type of container of unit size, the goal is to find the maximal size for a given set of *identical* items that can be packed into the container: consult, e.g., Castillo et al. [5] for a circle packing model with a unit square container, with reference to related model versions. The assumption of identical packing objects leads to a problem structure that has substantial symmetries: such structures can often be exploited, to find promising initial configurations—in some cases, even to find provably optimal packings.

Our article focuses on packing problems with a finite number of general (in principle, arbitrary) objects to be packed into various types of finite-size containers. Such models are often somewhat informally referred to as irregular packings. Irregular packing needs arise in many practical applications such as additive manufacturing, communication network design, crystallography, dashboard layout design, engineering design, facility dispersion, facility layout, facility location, fiber optic cabling (and other types of cabling) design, furniture making, garment and shoe manufacturing, material design, optimized (bin, cargo, container, pallet, vehicle) loading, and raw material cutting. Important and hard-to-handle additional considerations can be related to load configuration, balancing, loading preferences, sequencing, and scheduling aspects. Without going into further details here, consult, e.g., Alvarez-Valdes et al. [6], Bennell and Oliveira [7], and Scheithauer [8], with extensive reference lists.

Let us also mention some specific examples of OP applications. The contributing authors of the volume edited by Fasano and Pintér [9] discuss the packing of hazardous containers on ships, dynamic packings for datacenter resource management, the optimized packing of free-form objects in engineering design, non-standard packing problems arising in space engineering applications, cutting and packing problems with placement constraints, container loading using heuristics, the design of optimal LED streetlights, approximate packing approaches, robust designs for circle coverings of a square, the optimized packing of jobs in spatial scheduling, optimized object packings using quasi-Φ-functions, graph-coloring models, and metaheuristics for packing applications.

1.2. Technical Challenges

Object-packing problems—including irregular packings—are notoriously hard due to their combinatorial complexity, combined with the computational difficulty of enforcing non-overlap and containment constraints. Direct analytical approaches are not viable to solve such general problems to optimality (except in certain cases, for the smallest problem instances). The symmetry structure implied by packing identical objects is absent, and good-quality numerical solution "guesses" may not be readily available. Therefore, in order to find conjectured (approximate) numerical solutions, continuous and combinatorial optimization—typically in combination with problem-specific insight, rules, and heuristics—have become the tools of choice. To illustrate this point, consult, e.g., Dowsland et al. [10], Bennell and Oliveira [7,11], Fasano [12,13], Stoyan et al. [14–16], Leao et al. [17], and Oh et al. [18], with further examples cited later on in this work.

The joint consideration of continuous and combinatorial optimization aspects of packing problems can be handled by appropriately tailored mixed-integer linear or nonlinear optimization models and solution techniques. Scheithauer [8] discusses the fundamental concepts, modeling, and solution approaches to several standard allocation, cutting, and packing problems. The corresponding models are handled by various techniques such as mixed-integer linear programming, nonlinear programming, constraint programming, and problem-dependent heuristics.

1.3. Contributions

In this article, we present broad classes of irregular packing problems and discuss approaches to their numerical solution. We focus on packing problems in which all objects can be freely positioned within the container. For this reason, we omit the discussion of decisions related to positioning items along a (small) set of fixed directions, which would require the introduction of integer decision variables. First, we introduce a continuous nonlinear optimization modeling framework applicable to a broad range of OP problems. Next, we provide a concise review of algorithmic approaches to packing problems, focusing on irregular packings. We introduce and discuss several scalable packing problem classes and their implementation in optimization modeling environments. The numerical solution of the resulting models is frequently based on nonlinear optimization combined with problem-specific heuristics. We present illustrative numerical test results for several scalable model types. These numerical results were obtained using our (average capacity) personal computers running under Windows OS versions, with Intel processors, and with 16 to 32 Gigabytes of RAM. All results are available upon request.

2. Continuous Nonlinear Optimization

2.1. Problem Statement

The object configuration problems discussed in this article can be formulated by applying the following nonlinear programming (NLP) model framework.

NLP:

(1) Minimize $f(x)$ under the conditions $x \in D$.
The feasible set D is defined as

(2) $D := \{x : l_i \le x_i \le u_i \text{ for } i = 1, \ldots, n, g_j(x) \le 0 \text{ for } j = 1, \ldots, m\}$.

Here, $x = (x_1, \ldots, x_n) \in R^n$ denotes the n-vector of decision variables; we assume that the objective function $f : R^n \to R$ is continuous, without further specifications at this point. We also assume that the set of feasible solutions D is non-empty, closed, and bounded: D is defined by n pairs of finite box constraints $l_i \le x_i \le u_i$ for $i = 1, \ldots, n$, and m general functional constraints $g = (g_1, \ldots, g_m)$. The vectors $l \in R^n$, $u \in R^n$ define the component-wise lower and upper bounds of x; $g : R^n \to R^m$ is the m-vector of continuous function constraints. In the context of our discussion, we can assume that f and/or some components g_j of g are nonlinear functions. Note also that while in purely box-constrained optimization problems $m = 0$, in OP problems typically $m > 0$.

For completeness, we remark that binary variables $y \in \{0, 1\}$ can be formally handled by models with continuous variables, e.g., by adding the constraints $0 \le y \le 1$ and $y(1 - y) \le 0$. The added nonlinear constraint is nonconvex, thereby numerically inconvenient: here, we only wish to point out the formal handling of a binary variable by a continuous variable. Similarly, more general integer variables can be deduced into continuous variables following their binary decomposition. Therefore, continuous nonlinear optimization models formally cover the entire class of mixed-integer nonlinear programming models.

2.2. Global Optimization for Object Packings

Notice the absence of the frequently postulated model convexity assumptions: convexity would be implied, e.g., by requiring that f and all components $g_j, j = 1, \ldots, m$, are convex functions, leading to essentially unimodal problems (in terms of their unique optimum value). In the absence of such convexity guarantees, instances of models (1)–(2) could have multiple local optima: consequently, we are interested in finding the globally optimal solution(s)—as opposed to finding one of the local optima. In object configuration problems, model convexity cannot be assumed: the OP model examples discussed in this work are all non-convex.

Observe next that—by the classical extreme value theorem of Weierstrass—the postulated assumptions (D is compact, f is continuous) guarantee that the NLP model has a globally optimal solution set X^*. Each $x^* \in X^*$ is a globally optimal solution; $f^* = f(x^*)$ the global optimum value. The set X^* often—but not always—consists of a single point x^*: we think that this is the typical scenario in irregular object-packing problems without symmetries, while models with identical objects have many equivalent global solutions.

Generally speaking, OP problems are notoriously hard. Finding the solution set X^*—or finding just one global solution $x^* \in X^*$—of model (1)$-$(2) using a purely analytical solution methodology is not possible. Finding X^* numerically can also become very challenging, since—as a rule—the difficulty of solving larger model instances is rapidly increasing. Additional challenges arise due to the typically "complicated" feasible set D. To illustrate the latter aspect, note that the no-overlap constraints are often nonconvex, and that their number increases quadratically as a function of the number of packed objects. Given these potentially massive difficulties, a local scope search for the best configuration is clearly insufficient on its own. A global search strategy—if carried out efficiently—leads to high-quality feasible solutions, sometimes even to the true numerical global solution! These observations are also supported by our numerical results, based on solving various instances of hard packing problems, reviewed later in this article.

In recent decades, global optimization (GO) has become a well-established research area: there exists a large number of books and thousands of articles that are devoted to the subject of GO. Here we only refer to the Handbook of Global Optimization volumes edited by Horst and Pardalos [19] and Pardalos and Romeijn [20]. The key theoretical results regarding the most important GO model types and algorithmic solution strategies have been followed by software implementations that are used to handle a range of GO applications. Interested readers should consult, e.g., Pintér [21–23] for discussions with extensive references to GO algorithms, software, benchmarking, and applications.

The development and verification of robust and efficient GO software requires practically important and challenging test problems. The most widely used "classical" GO test problems are small-scale, either box-constrained models, or have only a handful of constraints: consult, e.g., Floudas et al. [24]. Hence, their numerical solution does not pose a serious challenge to state-of-the-art GO software on today's computing platforms. The scalable object-packing models discussed in this article are—or can become—useful addenda to GO test libraries.

3. A Review of Approaches to Irregular Object Packings

3.1. Algorithmic and Heuristic Approaches

We discuss these approaches together, since they are often utilized in problem-type dependent combinations.

Dowsland and Dowsland [25] offer an early survey on the application of Operational Research techniques to the solution of packing problems. Their survey is focused on the modeling and solution of practically motivated problems in two and three dimensions: both exact and heuristic solution approaches are reviewed. They point out the well-known fact that even the "simple" rectangular packing problem class is known to be "hard" (in technical terms, NP-complete). Therefore, it is often impossible to provide exact solutions for sizeable problems within a reasonable computational time. Two-dimensional rectangular packing, pallet loading, one- and two-dimensional bin packing, strip packing, and three-dimensional container-packing problems are discussed, with references. For handling the more difficult problem types, common sense-based insights and heuristic approaches are also utilized. In the case of packing non-rectangular objects, the approaches vary according to their geometry. The authors also note the potential of stochastic modeling and randomized solution approaches: an observation that remains valid for the case of handling irregular packings. Let us emphasize that irregular packings typically lead to even far more difficult problem types than the rectangular packings discussed by the authors.

Bennell and Oliveira [11] present a tutorial with the goal of covering the core geometric approaches and methods employed by researchers in the cutting and packing of irregular shapes. They note the need for geometric tools to handle the wide variety and complexity of shapes that need to be packed. The paper describes the operations of several different approaches for dealing with the geometry required in the solution of packing problems. The methods discussed include raster methods (which divide the continuous stock sheet into discrete areas), trigonometric analysis (of packed polygons), no-fit polygons (used to check whether two polygons overlap or not), and Φ-functions (used to represent all mutual positions of two objects, to decide whether they overlap or not). Let us add here that Bennell et al. [26] provide a review of some earlier studies that use Φ-functions, more recent applications will be referred to in this work.

Bennell and Oliveira [7] discuss irregular shape nesting (packing) problems. They emphasize the essential need for utilizing heuristic approaches. They also propose dividing the approaches into methods that work with partial solutions, building these up to a final design (constructive heuristics), and methods that work with complete proposed solutions where changes are made in order to find improvements (improvement heuristics). The order in which the pieces are selected for inclusion in a constructive heuristic can be based on fixed rules, dynamic selection, and randomization, with or without backtracking options. These choices can have a significant influence on the resulting (proposed) solution. Working with complete solutions requires the design and application of some efficient local search heuristics, which depend on the problem representation. The problem can be considered in its entirety (physical layout), or as a sequence of the objects to pack to arrive at the final layout. Moves can be based on swapping objects and inserting objects. The authors summarize their findings in an insightful diagram (cf. [7], Figure 18), which displays a proposed organization of nesting solution approaches.

Fasano [12] presents and discusses research carried out in support of the cargo accommodation of space vehicles. The typical goal of such studies is to maximize the loaded cargo, while also considering additional requirements. The packed items can often be modeled as parallelepipeds, but this approximation is frequently not acceptable. Additional considerations, such as load balancing, give rise to even more challenging non-standard packing problems. The article first considers the orthogonal packing of "Tetris-like" items within a convex domain, and then the packing of polygons with continuous rotations in a convex domain. The proposed solution approaches are based on mixed-integer linear or nonlinear programming (MILP, MINLP).

Fasano [13] summarizes the results of his research aimed at tackling non-standard packing issues arising in space engineering and logistics. In these applications, the necessity of exploiting the spacecraft load capacity, as much as possible, represents a paramount challenge. He proposes a global optimization framework based on MILP, MINLP, and heuristic strategies. His study offers insights related to possible applications across several engineering and industrial sectors, from transportation to manufacturing.

Jones [27] describes a general algorithm for nesting irregular shapes. The key idea is to inscribe a selection of circles in each irregular shape and then relax the non-overlap constraints for the shapes by replacing them with non-overlap constraints for the inscribed circles. A specialized branch-and-bound algorithm with added heuristics is introduced to find the initial inscribed circles that approximate the shapes. This work shows an interesting connection between general circle packings (with possible overlaps among the circles) and general irregular packings.

Leao et al. [17] review mathematical models for handling irregular packing problems. These hard nesting problems had been addressed earlier, both in the scientific literature and in real-world applications, applying heuristic and meta-heuristic techniques. More recently, a variety of mathematical models have been proposed for nesting problems. These models can be used either to prove optimal solutions for certain types of nesting problems or can frequently serve as the basis of problem type-specific heuristic approaches to improve the approximate solution found by optimization.

Pankratov et al. [28] consider packing irregular solid objects in a cuboid of minimum volume. Each considered object type is composed of a number of convex shapes, such as oblique and right circular cylinders, cones, and truncated cones. New analytical tools are introduced to state placement constraints for oblique shapes. Using the Φ-function technique originally proposed by Stoyan and subsequently utilized in studies with his colleagues, the packing problem is formulated as a nonlinear programming problem. The solution approach is illustrated by numerical examples.

3.2. An Algorithm Framework for Object Packings

Based on the literature reviewed and the preceding discussion, a conceptual framework for developing object-packing algorithms can be summarized as follows (obviously, implementations will be problem-specific).

1. Create an initial feasible solution, or—more generally—a set of candidate solutions. It is possible to build candidate solutions sequentially (adding objects piecewise) or to create complete candidate solutions. Both deterministic and stochastic approaches can be used to find such initial packing configurations. The availability of feasible solutions leads

to establishing bounds related to the quality of the current best solution, which can be compared to estimates of the best possible solution (simple estimates are often easy to find). Let us point out that finding credible solution estimates can be straightforward in certain model types: notable cases are some scalable models discussed later on in this work.

2. Using the initial feasible solution(s), solve numerically the resulting optimization model. Depending on the model type considered, solution approaches may involve global or local NLP with continuous decision variables, integer programming (IP) with discrete decision variables, or mixed variable (MILP and MINLP) models. In all these cases, finding high-quality feasible and near-optimal solutions to packing models typically requires global scope search techniques. Since it can be very hard to solve OP models to global optimality, the solution obtained is typically only an approximation of the optimal solution.

3. In order to improve solution quality, one can apply various heuristic approaches. Such methods can be based on meta-heuristic algorithms that help find approximate solutions to complex optimization problems, with a computational effort that is deemed reasonable. Examples of heuristic methods that can be put to good use in OP are basin hopping, differential evolution, genetic algorithms, greedy randomized adaptive search, neighborhood search, simulated annealing, swarm intelligence algorithms, and tabu search. Problem-specific insight can motivate object sequencing, swapping, backtracking, and other strategies, which become part of tailored meta-heuristic algorithms. A simple but generally valid observation is that applying more insightful and sophisticated heuristics can assist in finding higher-quality solutions to hard OP problems.

4. While executing stages 2 and 3, one can always compare the current candidate solution(s) to the estimated best solution (we will comment later on finding solution estimates). Such comparisons can be effectively used to establish algorithmic search termination criteria. Obviously, this is most useful since in OP problems with continuous decision variables the global scope search for the best solution theoretically could go on infinitely, while in discrete variable spaces, the solution effort is subject to the "curse of dimensionality".

In addition to the cited OP-specific literature, we refer to three relevant volumes on heuristics: Martí et al. [29], Gendreau and Potvin [30], and Taillard [31]. There are many other works devoted to this very useful (and very fashionable) topic.

Next, we illustrate the application of various algorithmic approaches by numerically solving instances from several OP model types.

4. General Circle Packings

4.1. Problem Statement

The general circle packing (GCP) problem discussed here is defined as follows. Given N (in principle, arbitrary size) circles with radii r_i for $i = 1, \ldots, N$, find the minimum radius r_0 of the circular container that contains these N circles in a non-overlapping arrangement. Assuming that the container is centered at the origin, we introduce the decision variables (x_i, y_i) for the center of circle i, in addition to r_0. Denote by $e_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ the Euclidean distance between the centers of circles i and j. Then, a possible model formulation is

GCP:

(3) Minimize r_0

(4) $x_i^2 + y_i^2 \le (r_0 - r_i)^2 \quad$ for $i = 1, \ldots, N$

(5) $(r_i + r_j)^2 \le e_{ij}^{\;2} \quad$ for $i, j = 1, \ldots, N, i < j$.

The GCP model has $2N + 1$ decision variables, N convex constraints (see (4)), and $N(N-1)/2$ nonconvex constraints (see (5)). The GCP model is an example of a *scalable* object-packing model class: for a given N and given sequence of radii r_i for $i = 1, \ldots, N$, we can directly create the corresponding model instance.

4.2. GCP Literature Review

The GCP problem—including the special case in which all circles have the same radius—has been studied by many researchers over several decades: some of these works are cited next. We mention first the book by Szabó et al. [32] which presents an introduction and history to the problem of packing equal circles into a square and some other containers, with further packing examples. This work discusses both exact methods and stochastic algorithms that can be applied or generalized to address packing problems.

Next, we highlight a few representative articles devoted to handling GCP problem instances, using exact and heuristic methods. All cited works demonstrate the merits of their proposed strategies, in terms of solution quality and computational efficiency on specific model instances. As noted earlier, numerical difficulty rapidly increases as N increases, and in most cases, only tentative ("best known") solutions are presented. This comment also applies to the forthcoming—more difficult—model classes discussed here.

In one of the earliest GCP studies, George et al. [33] discuss the problem of packing circles into a rectangle using heuristic rules in different combinations. Due to the binary consideration of whether to include a circle in the packing or not (after assigning priority "weights" to each circle), they formulate an MINLP problem and propose heuristic procedures to approximately solve it. Test models with up to 40 circles are based on the intended application of the study (fitting pipes of different diameters into a shipping container).

Pintér and Kampas [34] propose the use of "pure" global optimization—without heuristic enhancements—to test its applicability to selected identical CP and GCP problems. For the GCP case, illustrative results are presented for packing 20 circles (without claiming global optimality). They note the potential benefits of utilizing structural considerations and heuristic initial arrangements.

Huang et al. [35] propose two new heuristics to solve the GCP problem. The first one is a core heuristic, which selects the next circle to place according to the maximal hole degree rule. The second heuristic uses a look-ahead strategy to improve the first one.

Addis et al. [36] combine monotonic basin hopping, randomized search, population-based diversification, and local optimization to solve a given set of GCP problems. Basin hopping (Wales and Doye [37]) is a global scope heuristic optimization technique that iterates by performing random perturbations of candidate solutions, followed by efficient local optimization. It is a particularly useful approach to global optimization in high-dimensional landscapes with many close-to-optimal solutions, such as finding the minimum energy structure for molecules, or—in the present context—solving irregular packing problems. Addis et al. also apply variable space reduction by initially discarding the smallest circles to pack. Their combination of exact and heuristic approaches led to the best overall solution set for the Zimmermann GCP competition [38], based on a given set of model instances with up to 50 circles.

To illustrate the point of finding ever-improving tentative solutions for model instances, note that years after the GCP competition—which drew the attention of many expert researchers—improved best known solutions have been found, as documented by Specht [3]. This remark is also pertinent regarding the other model classes presented here, which are substantially more difficult than the GCP model class.

Castillo et al. [5] discuss several circle-packing problem types with their industrial applications and refer to exact and heuristic strategies for their solution. They review related earlier research and present illustrative numerical results using global optimization software packages. The results obtained are improved by implementing an *a posteriori* refinement strategy that—based on a high-quality initial circle configuration—swaps all pairs of adjacent-sized circles until no further local improvement is possible. They report close-to-best-known numerical results for a GCP test problem class with up to 35 circles.

Al-Mudahka et al. [39] present an adaptive algorithm that incorporated nested partitioning within a tabu search framework and apply diversification strategies to approximate the global optimum in GCP problems. The tabu search component serves to identify the ordering of the packed circles, then nested partitioning is applied to determine the circle positions.

Specht [40] proposes a method to detect voids in general circle packing configurations. He notes that smaller circles often get stuck in trapped positions. Hence, the knowledge of the structure of unoccupied areas or holes inside a packing is important to be able to move trapped circles into free circular places or voids. The proposed algorithm for detecting such voids in two-dimensional circle packings is based on a decomposition of the contact graph of the circles. Combined with heuristic methods like object jumping, swaps, and shifts, this approach can increase the solution quality significantly.

Romanova et al. [41] consider the GCP problem under additional balancing and distance conditions, arising in the context of 3D printing. Two problems are studied: the first minimizes the container's radius, while the second maximizes the minimal distance between circles, as well as between circles and the boundary of the container. Mathematical models and solution strategies are presented and illustrated with computational results.

Without going into further details on the GCP problem and its variants, we refer to several other studies that combine nonlinear optimization with various heuristics: consult, e.g., Grosso et al. [42], Lü and Huang [43], Hifi and M'Hallah [44], López and Beasley [45], Ryu et al. [46], He et al. [47], and Stoyan et al. [16].

4.3. Solving GCP Instances: Illustrative Results

Optimization model development environments can be used to efficiently formulate and solve scalable object configuration problems. Scalable models require only some input parameter changes (in GCP, the number of circles to pack and their sizes), in order to create new instances. Typically, a range of global and local NLP solver engines are linked to the leading optimization model development environments. This allows flexible solver option changes and directly supports the comparative assessment of solver capabilities for a given set of models. In our OP studies discussed here, we have been using the optimization model development environment AMPL [48] and the scientific-technical computing system *Mathematica* [49] with several available solver options.

The GCP model class is the simplest among the scalable OP models considered here. However, it already poses a numerical challenge to solvers—even for the smallest model instances. The results presented below illustrate this point, based on using six solver options (CONOPT, IPOPT, LGO, LOQO, MINOS, and SNOPT) linked to AMPL.

Let us note here that the solver LGO is capable of determining high-quality numerical solutions to global optimization problems that have (possibly many) locally optimal solutions: for theoretical background and implementation details, consult [50–53]. The other solvers CONOPT, IPOPT, LOQO, MINOS, and SNOPT are widely used local optimization solvers: for summary information, consult, e.g., the AMPL website [48].

Table 1 includes the best-known results retrieved from [3], for instances of the GCP model class with $r_i = i$, for $i = 1, \dots, N$, $N = 5, \dots,$ 10. The rate of deviation of the solver solution f^s from the best-known solution f^* is defined as $\frac{f^s}{f^*}$: for high-precision solutions, this rate is close to 1, while larger values indicate less precise numerical solutions.

Table 1. Comparison of solver performance on small GCP model instances.

No. of Circles	**5**	**6**	**7**	**8**	**9**	**10**
Best known	9.001397746	11.05704040	13.46211068	16.22174668	19.23319391	22.00019301
conopt	9.001397746	11.27687246	13.69914165	16.89847646	19.55248862	24.27842228
ipopt	9.001397671	11.53266465	13.69914153	16.63722613	20.15497158	23.37559224
lgo	9.001397749	11.05704040	13.69914165	16.22174668	19.76606329	22.55237926
loqo	9.001397757	11.53266476	13.69914168	16.63722632	20.15497177	23.38725283
minos	9.001397746	11.53266474	13.69914165	16.44050689	20.27681461	23.42606205
snopt	9.001397746	11.53266474	13.69913854	16.63722700	20.15497175	23.42606205

Table 1. *Cont.*

No. of Circles	5	6	7	8	9	10
Deviation from best known						
conopt	1.0000000	1.0198816	1.0176073	1.0417174	1.0166012	1.1035550
ipopt	1.0000000	1.0430155	1.0176073	1.0256125	1.0479264	1.0625176
lgo	1.0000000	1.0000000	1.0176073	1.0000000	1.0277057	1.0250992
loqo	1.0000000	1.0430155	1.0176073	1.0256125	1.0479264	1.0630476
minos	1.0000000	1.0430155	1.0176073	1.0134856	1.0542614	1.0648117
snopt	1.0000000	1.0430155	1.0176070	1.0256126	1.0479264	1.0648117

As these results indicate, all solvers easily handle the smallest GCP model instance included here, but thereafter, all the local solvers fail to find the best solution. LGO, a global scope solver—used here without any added heuristics—fares better than the local solvers: in default usage mode, LGO finds the global solution in three out of six cases, but it misses the global solution by 1.8% to 2.8% in the other three cases.

Table 1 merely serves to illustrate the optimization challenge implied by the proposed model class, noting again that GCP is the simplest model class discussed here. The runtimes for the local solvers—including LGO when used in local solver mode—for each of these small-scale problems are just a fraction of a second. The LGO runtime—when used in global solver mode—is 12.53 s for the largest model (with $N = 10$) included above.

To visualize the numerical solution of a GCP model, see Figure 1 below, which corresponds to the 10-circle model instance from the Zimmermann GCP challenge. To handle GCP models, we also used LGO linked *to Mathematica* with the product name *MathOptimizer Professional* [53]. All visual illustrations in this article were produced using *Mathematica*.

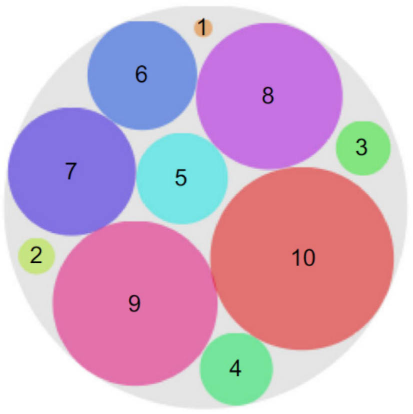

Figure 1. A solved GCP model instance.

5. Further Scalable Object Configuration Problems

5.1. Scalable Models

Conceptually, scalable object packings can be characterized by postulating that all packed objects have a similar or identical geometric structure—while their sizes and shapes still are or can be different to some extent, depending on the type of the objects. For example, one can think of collections of circles, spheres, ellipses, or ovals to pack: next, we will present models for the last three types of problems.

To create scalable optimization models of interest, we have to guarantee by suitable model parameterization that the resulting models do not become "too simple" to handle. Although infinite tessellations created in finite regions can have wonderful aesthetic value (think of some historical mosaic designs or modern works by M.C. Escher and followers), here we want to define finite packings, which become unbounded in size as the number of objects increases. For example, in the GCP model class, we can consider a collection of circles with radii $r_i = i^{1/2}$, or $r_i = i$ for $i = 1, \ldots, N$ to be packed into a corresponding container circle. In both of these cases, the size of the corresponding optimal container will approach infinity, as $N \to \infty$.

Without going into further details, we remark that properly combined scalable OP models directly lead to new scalable OP models. As an example, think of packing finite collections of object types—each defined by iterative formulas—into a container.

In scalable OP problems with objects having the same basic geometry (such as collections of circles, or spheres, ellipses, ovals, triangles, rectangles, polygons, etc.), the number of decision variables that serve to jointly describe the object positions is typically a *linear* function of the number of objects considered. The number of constraints is partially dictated by the pairwise non-overlapping criterion: the number of such constraints increases quadratically with the number of packed objects. Additional variables and constraints may be present, depending on the problem definition and structure. The GCP model class can serve to illustrate the above observations.

For conceptual clarity, we refer to the typology of cutting and packing problems suggested by Wäscher et al. [54], which facilitates the organization and categorization of the OP literature. In terms of their problem classification, the GCP problem class and each of the scalable model classes presented here are strongly heterogeneous assortment problems with non-identical packed items and non-orthogonal layouts with free rotations allowed, where one container is considered with a single objective (volume minimization) in the presence of additional no-overlap and strict containment constraints.

To illustrate further scalable OP model classes, next, we discuss specific cases and examples based on our related studies [5,55–60]. While [55] is a direct extension of the GCP model, the three other model-types discussed are more complicated to describe: therefore, for these models, only concise problem formulations are presented and illustrated here, with reference to our related works.

For further examples of irregular OP problems with numerical solutions, consult, e.g., Fasano [12,13], Pankratov et al. [28], Romanova et al. [41], Duriagina et al. [61].

5.2. General Sphere Packings in R^d

This model type is a direct generalization of the GCP problem for arbitrary dimension $d > 2$. Given a finite collection of d-dimensional spheres, our goal is to find the smallest sphere in R^d that contains the given spheres in a non-overlapping arrangement. For $i = 1, \ldots, N$, let $S_i \subset R^d$ denote a d-sphere with radius $r_i > 0$. Let S_0 denote the container d-sphere with radius r_0. Similarly to the GCP model, we set $c_0 = 0 \in R^d$ as the center of the container sphere and denote the (to be optimized) center position of sphere S_i by $c_i = \{x_{i,1},\ x_{i,2}, \ldots,\ x_{i,d}\}$. The Euclidean distance between the pair of d-sphere centers c_i and c_j is denoted by $e_{ij} = ||c_i - c_j||$, where $||c_i|| = \sqrt{\sum_{k=1}^{d} x_{i,k}^2}$.

Applying this notation, the general sphere packing (GSP) model is formulated as follows.

GSP:

(6) Minimize r_0

(7) $||ci||^2 \le (r_0 - r_i)^2 \quad \text{for } i = 1, \ldots, N$

(8) $(r_i + r_j)^2 \le e_{ij}^{\;2} \quad \text{for } i, j = 1, \ldots, N, i < j.$

To illustrate the numerical solution of GSP instances, below we present optimized sphere configuration results with up to $N = 50$ spheres with radii r_i for $i = 1, \ldots, N$ in dimension $d = 3$. In our study, we used *Mathematica* and LGO linked to *Mathematica* on one of our personal computers. For comparison, we also present results using a hybrid heuristic optimization approach. For further details, consult Pintér et al. [55].

For $d = 2$, our numerical results are, on average, within 1% of the entire set of best-known results for the Zimmermann [38] model instances in R^2. We found new (conjectured) sphere packings for previously unexplored generalizations of the same model class in R^{d} with $d = 3, 4, 5$, as can be seen in the examples in Table 2. As an example, Figure 2 displays the numerically optimized 10-sphere instance in R^3.

Table 2. Sphere packings in R^d with $d = 3, 4$: illustrative results.

		LGO Result		**Hybrid Result**	
$\boldsymbol{R^d}$	$\boldsymbol{N}$	**Objective** $\boldsymbol{r_0}$	**Time (s)**	**Objective** $\boldsymbol{r_0}$	**Time (s)**
R^3	10	19.5361	2.1	19.5361	23
	15	31.1456	6.6	31.3662	51.6
	20	44.8945	22.9	44.4319	172.5
	25	59.5338	73.6	58.7339	293.4
	30	74.9691	202.4	74.2992	232.9
	50	144.317	4231.9	142.834	1360
R^4	10	18.924	2.3	18.8575	22.9
	15	30.4039	8.2	30.4039	34.6
	20	41.4775	28.4	41.4858	74.3
	25	53.8574	95	53.5299	83.7
	30	66.5128	246.7	66.6449	102.7
	50	121.820	5056.9	121.806	552.6

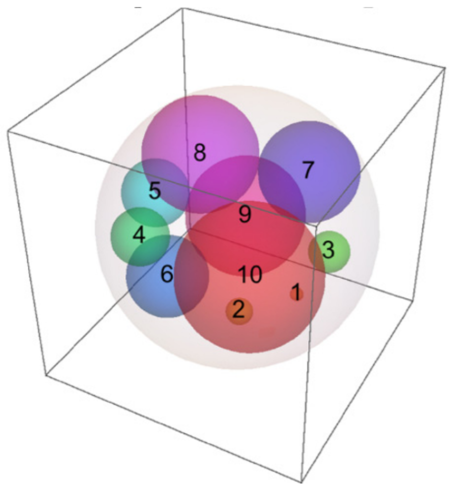

Figure 2. A solved GSP model instance.

Without going into further details, let us point out that—similarly to generalizing GCP to GSP—all three model classes considered next can be directly generalized to arbitrary dimension $d > 2$, with corresponding objects and containers defined in R^d.

5.3. General Ellipse Packings in R^2

Given a set of general ellipses $i = 1, \ldots, N$ in R^2, our goal here is to find their packing into an optimized regular polygon. We will refer to this class of problems as GEP models. Specifically, for a given set of ellipses with arbitrary size and orientation, and a given integer $M \geq 3$, our objective is to minimize the apothem (the line segment from the center of the polygon to the midpoint of one of its sides) of the regular M-polygon container. Therefore, packing ellipses into a regular polygon requires i) the determination of the maximal distance from the center of all polygon faces to each ellipse boundary (in order to contain all ellipses), and ii) the finding of the minimal distance between all pairs of the ellipses (in order to avoid ellipse overlaps as a function of ellipse center locations and orientations). The first requirement is necessary to determine the length of the polygon's apothem α, which is then to be minimized. The second requirement serves to prevent the ellipses from overlapping. Explicit analytical formulas for the first requirement can be directly derived. However, for the second requirement, deriving explicit analytical formulas would be complicated. Therefore, the ellipse-packing model class is based on embedding optimization calculations, using Lagrange multipliers λ, into the overall optimization strategy.

We summarize the key modeling steps below, referring to Kampas et al. [57] for details. Omitting index i for simpler notation, equation $e(a, b, x^c, y^c, \theta; x, y) = 0$ defines the boundary of an ellipse with semi-major and semi-minor axes a and b, centered at $\{x^c, y^c\}$ and rotated counterclockwise by angle θ. The value of $e(a, b, x^c, y^c, \theta; x, y)$ is negative for all points (x, y) located inside the ellipse, zero for all points on the ellipse boundary, and positive for all points outside the ellipse. Here, $\{x^c, y^c\}$ and θ are the primary decision variables for each ellipse i: these variables are denoted by $\left(x_i^c, y_i^c\right)$ and θ_i for $i = 1, \ldots, N$. All pairs of packed ellipses are prevented from overlapping by requiring that the minimum value of the ellipse equation for the first ellipse (ellipse i) for any point on the second ellipse (ellipse j) must be greater than a sufficiently small parameter $\epsilon \geq 0$ (this way, $e(a, b, x^c, y^c, \theta; x, y)$ is used to define the Φ-function, referred to earlier, for pairs of general ellipses). This non-overlapping requirement between ellipses i and j is met utilizing the embedded Lagrange multipliers λ_{ij} using partial derivatives of the ellipse equation.

In terms of scalability, ellipse-packing problem instances could be generated, e.g., by defining the following input structure: $a_i = i^{-1/2}$, $b_i = a_i/c$ for $i = 1, \ldots, N$, where $c > 0$ denotes the eccentricity of the ellipses. Note that with $c = 1$, this problem becomes a GCP problem for circles with radii $a_i = 1/\sqrt{i}$. In [57], *MathOptimizer Professional* as well as IPOPT linked to *Mathematica* were used as solvers. IPOPT is the COIN-OR Interior Point Optimizer [62], based on research by Wächter and Biegler [63]. The IPOPT link to *Mathematica* is documented at [64].

In Table 3, we present illustrative results for the above-mentioned scalable model class with the parameters $c = 2$ and $M = 5$, 10 (corresponding to regular pentagon and decagon containers). See Figure 3 for a solved GEP model instance.

Table 3. Illustrative GEP model results.

Container Sides	N	Objective α	Container Area	Time (s)
$M = 5$	5	1.3535	4.3560	22.1
	6	1.3909	4.6001	29.7
	7	1.4281	4.8492	40.8
	8	1.4614	5.0780	52.6
	9	1.4904	5.2815	72.7
	10	1.5118	5.4340	85.8
$M = 10$	5	1.1979	4.2170	79.4
	6	1.2422	4.5350	106
	7	1.2777	4.7978	134
	8	1.2974	4.9472	178
	9	1.3255	5.1635	224.4
	10	1.3481	5.3408	294.7

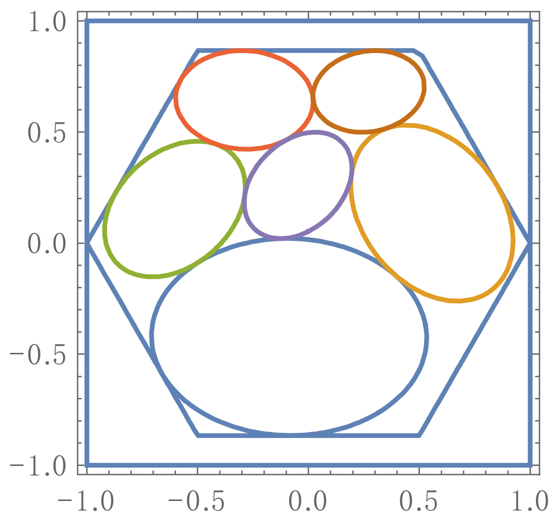

Figure 3. A solved GEP model instance.

For further details and numerical results, consult Kampas et al. [56] where we study ellipse packings in an optimized circular container, and Kampas et al. [57] where we discuss the case of optimized regular polygon containers.

5.4. General Oval Packings in R^2

Given a set of generalized ellipses (egg-shaped objects), referred to here as ovals, our goal is to find their packing into an optimized regular polygon in R^2. We will refer to this class of problems as GOP models. As above, the objective is to minimize the area of the regular polygon that contains a given collection of ovals with arbitrary size and orientation.

The extension from packing ellipses to packing ovals is not trivial. Here we provide only a brief summary, referring to Kampas et al. [58] for details. The perimeter of an oval is defined by $(x/a)^p + e^{tx}(y/b)^p - 1 = 0$, where (x, y) denotes the location of a point on the oval perimeter, parameters $a > 0$ and $b > 0$ are the semi-major and semi-minor axes of the oval, $p \geq 2$ is an even integer, and $t \geq 0$ is a distortion factor. In general, an oval can be defined with arbitrary size and orientation parameters, noting that the size and distortion factors are constrained, in order to maintain the oval's convexity. The input parameters to define an optimization problem instance are the number of sides for the container and the semi-major axes, semi-minor axes, exponent, and distortion factor for each oval to be packed.

The primary decision variables are the polygon's apothem and the center position and orientation of the packed ovals. There are two sets of secondary variables. The first set consists of the positions of the distance-maximizing lines pointing from each oval boundary to the center of each of the polygon faces. The second set is given by the positions of the points on one of each pair of ovals, which minimizes the value of the equation describing the other oval. These secondary variables are used to define the model constraints. Specifically, the first set of secondary variables is used to represent the constraints that keep the ovals inside the container. The second set of secondary variables is used to prevent the ovals from overlapping. These constraint sets are also generated by embedded Lagrange multiplier conditions.

Omitting index i (for simpler notation), equation $e(a, b, p, t, x^c, y^c, \theta; x, y) = 0$ defines the perimeter of an oval centered at $\{x^c, y^c\}$ and rotated counterclockwise by angle θ. As before, together with the apothem α, (x^c, y^c) and θ are the primary decision variables for each oval. All pairs of packed ovals are prevented from overlapping by requiring that the minimum value of the oval equation for oval i for any point on oval j has to be greater than a sufficiently small parameter $\epsilon \geq 0$. This non-overlapping requirement between ovals i and j will be met using embedded Lagrange multipliers λ_{ij} using partial derivatives of the oval equation. Let us point out that the slope of oval curve i equals the slope of oval curve j at the point on oval j that minimizes or maximizes the value of the function describing oval i; thus, the non-overlapping constraints emerge naturally from these conditions.

In terms of defining scalable models, oval packings (with their flexible parameterization options) offer tremendous potential. The table below summarizes some oval-packing test problem sets that could be used with M-polygon containers $M \geq 3$. Note that by setting $p = 2$, $a_i = b_i$, and $t = 0$, test case 1 becomes a general circle-packing problem for circles with radii $a_i = i^{-1/2}$. With $p = 2$, $a_i > b_i$, and $t = 0$, test case 2 becomes a general ellipse-packing problem for ellipses with semi-major and semi-minor axes, $a_i = i^{-1/2}$, $b_i = a_i/c$ with eccentricity $c = 2$. Test cases 1–6 and 8 consider ovals with the same distortion factor set ($t = 0$, 0.5, or 1). Test case 7 considers ovals with different distortion factors $t_i = i/5$.

To solve the model instances shown in Table 4, IPOPT linked to *Mathematica* was used as the solver. Here, we present GOP results for test case 8 and $M = 5,\ 10$ (regular pentagon and decagon containers), see Table 5. The increasing runtimes required to find visibly good solutions clearly indicate the challenge posed by this model class. Kampas et al. [57] include detailed results and images of optimized configurations for hundreds of instances in total from the four model classes presented here. See Figure 4 for a solved GOP model instance.

Table 4. Illustrative GOP model instances.

Test Case	p_i	(a_i,b_i)	t_i
1	2	$\left(i^{-1/2}, a_i\right)$	0.0
2	2	$\left(i^{-1/2}, a_i/2\right)$	0.0
3	2	$\left(i^{-1/2}, a_i\right)$	0.5
4	2	$\left(i^{-1/2}, a_i/2\right)$	0.5
5	2	$\left(i^{-1/2}, a_i\right)$	1.0
6	2	$\left(i^{-1/2}, a_i/2\right)$	1.0
7	2	$\left(i^{-1/2}, a_i\right)$	$i/5$
8	4	$\left(i^{-1/2}, a_i\right)$	0.0

Table 5. Illustrative GOP model results.

Container Sides	n	Objective α	Container Area	Time (s)
$M = 5$	5	2.2389	11.9184	127.4
	6	2.2976	12.5509	189.9
	7	2.3291	12.898	299.6
	8	2.3904	13.5858	312.7
	9	2.5637	15.6268	753.6
	10	2.7022	17.3613	757.5
$M = 10$	5	2.0385	12.2126	210.2
	6	2.0892	12.8277	351.3
	7	2.1804	13.9716	437
	8	2.3291	15.9423	539
	9	2.3294	15.9465	888.4
	10	2.4129	17.1113	1493

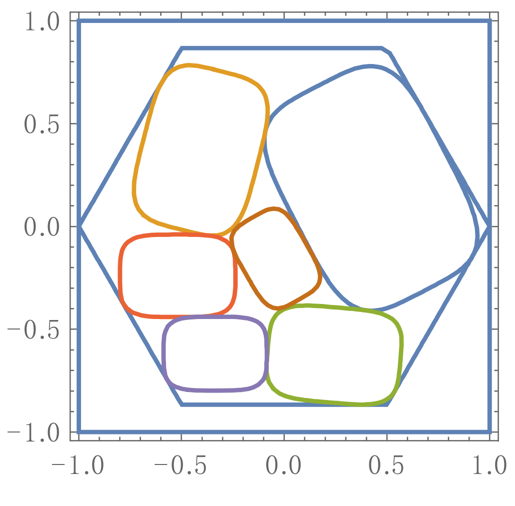

Figure 4. A solved GOP model instance.

5.5. P-Dispersion Configurations of Oval Objects in R^2

In this problem statement, we consider allocating "sizeable" (in other words, area-consuming) heterogeneous objects within a given feasible region in R^2. All objects are modeled by general ovals, which cover circles and ellipses as special cases. The feasible region could be convex (modeled here by regular polygons) or non-convex (modeled here by the intersection of general ovals). Our objective is to find optimally dispersed configurations, by maximizing the minimal separation between the boundaries of the oval objects and the boundary of the feasible region instances. For brevity, we refer to this model class as PDO (p-dispersion *with* ovals). Here we provide a summary of our modeling and solution approach, referring to Castillo et al. [60] for further details.

In [58], embedded Lagrange multiplier conditions have been used to produce optimized oval configurations with the objective of minimizing the area of the feasible region that contains a given collection of oval objects since our goal was to produce dense object configurations. Now, using a similar Lagrangian framework, our aim is to produce optimally dispersed configurations by maximizing the separation between the boundaries of the oval objects and the boundary of the feasible region. A pair of oval objects is maximally dispersed by maximizing the minimum value of the oval equation for object i for any point on the oval equation for object j, achieved by using embedded λ_{ji} Lagrange multiplier conditions. We note that in the dispersion case considered here, the Lagrange multipliers λ_{ji} must be negative to obtain the minimum: this requirement. with respect to the sign of λ_{ji}, is enforced by appropriately setting variable bounds during the optimization.

In the numerical examples presented here, a heuristic global optimization strategy (based on selecting multiple starting points) is used: we use these randomized initial solutions followed by calls to the local solver IPOPT linked to *Mathematica*. Three key parameters guide our optimization strategy: the number of random starts, the solver seed, and the maximum number of solver iterations (for clarity, these are not IPOPT internal iterations; the maximum number of solver iterations takes the final result of one solver iteration and uses it as the input for the next solver iteration). Our starting solution assigns $\left(x_i^c, y_i^c\right)$ and $\left(x_{ij}, y_{ij}\right)$ uniformly distributed random values with bounds that are chosen appropriately for each model instance; assigns θ_i uniformly distributed random values in $[0, 2\pi]$; and assigns λ_{ij} uniformly distributed random values in $[-100, 0]$. Setting the solver seed parameter offers some control over the generation of the random starting points, and it also supports the generation of identical results in repeated runs if needed.

Table 6 summarizes a set of parametric object dispersion test cases considered. Note that by setting $p = 2$, $a_i = b_i$, and $t = 0.0$, test case 1 serves to disperse a number of circular objects with radii $a_i = 0.25r \cdot i^{-1/2}$. With $q = 2$, $a_i > b_i$, and $t = 0.0$, test case 2 serves to disperse ellipsoidal objects with semi-major axes $a_i = 0.25r \cdot i^{-1/2}$ and semi-minor axes $b_i = 0.6a_i$. For brevity, the distortion factor is not parameterized, and test cases 1–6 consider oval objects with the same distortion factor set to $t = 0.0$, 0.5, or 1.0. Finally, we also illustrate that different feasible regions could be considered; for instance, a square, a regular hexagon, a non-convex curved boundary formed in the interior of four intersecting ovals, or a non-convex polygon formed in the interior of six intersecting polygons. In general, test cases 1 and 2 (dispersing circular and ellipsoidal objects with no distortion) require less computational effort than the other cases (dispersing distorted oval objects). Also, elongated objects seem to require more computational effort, particularly test case 6 (dispersing rounded rectangular objects). On average, a square feasible region requires less computational effort than a hexagonal feasible region. Table 7 summarizes a set of illustrative results, cited from [60]. See Figure 5 for two solved PDO model instances.

Table 6. Illustrative PDO model instances.

Test Case	p_i	(a_i, b_i)	t_i
1	2	$\left(0.25r \cdot i^{-1/2}, a_i\right)$	0.0
2	2	$\left(0.25r \cdot i^{-1/2}, 0.6a_i\right)$	0.0
3	2	$\left(0.25r \cdot i^{-1/2}, 0.6a_i\right)$	0.5
4	2	$\left(0.25r \cdot i^{-1/2}, a_i\right)$	1.0
5	2	$\left(0.25r \cdot i^{-1/2}, 0.6a_i\right)$	1.0
6	4	$\left(0.25r \cdot i^{-1/2}, 0.6a_i\right)$	1.0

Table 7. Illustrative PDO model results.

		Square Region		Hexagonal Region	
Values of p, r	**Test Case**	**Objective**	**Time (s)**	**Objective**	**Time (s)**
$p = 3, r = 1.5$	1	0.021305	8.4	0.028127	12.8
	2	0.070798	10.1	0.090017	13.4
	3	0.072242	15.2	0.090042	28.5
	4	0.023848	20.7	0.029633	28.1
	5	0.073014	17	0.090184	23.4
	6	0.065005	85.6	0.070549	78.5
$p = 10, r = 1.0$	1	0.025543	212.6	0.027923	228.4
	2	0.048605	277	0.052259	253.4
	3	0.0492	398	0.05289	520.6
	4	0.028027	347.7	0.029664	540.4
	5	0.050149	386.3	0.054299	515.5
	6	0.040088	1617.8	0.043147	2186.6

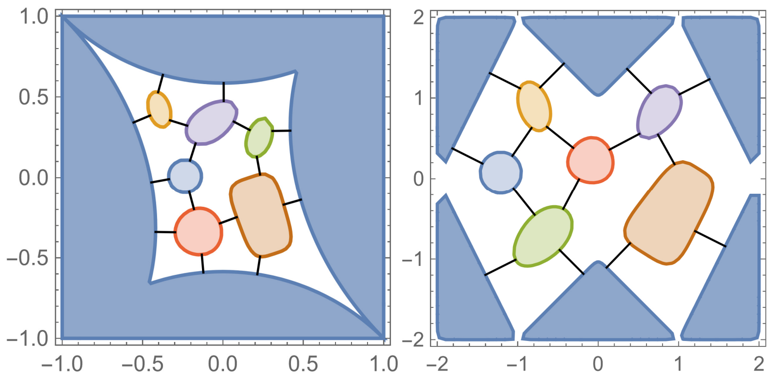

Figure 5. Two solved PDO model instances.

All numerical results reported here or presented in our cited articles are available from the authors.

5.6. Estimating the Quality of Numerical Solutions

In practical object-packing applications, a well-crafted, global scope search-based solution that incorporates problem-specific insight is typically quite acceptable. The illustrative results visually presented here and in our referenced works support this claim. A satisfactory numerical solution does not necessarily imply, however, that the solution found is within a guaranteed distance to the unknown best possible solution of some hard packing problem, even in a probabilistic sense (our preceding discussion related to the "simplest" GCP models illustrates this point). Therefore, estimating the quality of conjectured solutions is a relevant issue that we briefly address here.

A simple "brute force" approach to obtaining solution estimates can be based on solving the same packing problem a number of times using some randomization mechanism and recording the best solution found as an estimate of the unknown solution. This general approach can be theoretically validated by standard stochastic convergence arguments (assuming that the sampling procedure is theoretically exhaustive in the decision variable space, and that the sample size can be arbitrarily increased as required). Randomized global optimization methods have been extensively studied: consult, e.g., Pintér [50], Zabinsky [65], Zhigljavsky and Žilinskas [66].

Sophisticated extensions of the outlined global sampling approach can be based on applying the theory of extreme-order statistics and utilizing the sampling distribution of the optimum estimates. For details, we refer to Zhigljavsky and Žilinskas [66], de Haan and Ferreira [67].

In certain scalable models, optimality bounds can be based on considering the sequence of solved model instances (assuming the availability of sufficiently high-quality solutions). Let us also note that for certain scalable model types, regression models can be developed that support the estimation of the optimum value. This becomes especially useful for estimating the optimum value in higher-dimensional model instances that may be intractable by currently available software. To illustrate this approach, we refer to Pintér et al. [55].

6. Concluding Remarks

The universe of *nonlinear* systems and processes provides a limitless source of model development and optimization challenges: many of these problems require a global scope optimization (GO) approach. The structural properties of practical GO challenges—including the object-packing problems discussed here—frequently support the creation of "reasonable" initial solutions. However, finding—or just numerically approximating—the globally best solution remains a challenge that is not going away, in spite of the ever-increasing computing power at hand.

In this study, we discuss five scalable general packing model classes: circle packings (GCP), higher-dimensional sphere packings (GSP), ellipse packings (GEP), oval packings (GOP), and p-dispersion models with oval objects (PDO). Our numerical experiments illustrate the non-trivial nature of these model classes. Solution difficulty often dramatically increases as the model instance size increases. Insightful modeling and the skillful combination of generic nonlinear optimization methods with heuristic strategies are essential in order to find credible, highly optimized numerical solutions.

As a "side-benefit" of similar studies, we argue that scalable irregular object-packing models can also become a useful source of exceptionally hard GO test challenges.

Author Contributions: Conceptualization, J.D.P., I.C. and F.J.K.; methodology, J.D.P., I.C. and F.J.K.; software, J.D.P., I.C. and F.J.K.; validation, J.D.P., I.C. and F.J.K.; formal analysis, J.D.P., I.C. and F.J.K.; investigation, J.D.P., I.C. and F.J.K.; resources, J.D.P., I.C. and F.J.K.; data curation, J.D.P., I.C. and F.J.K.; writing, J.D.P., I.C. and F.J.K.; visualization, J.D.P., I.C. and F.J.K.; supervision, J.D.P., I.C. and F.J.K.; project administration, J.D.P., I.C. and F.J.K.; funding acquisition, J.D.P., I.C. and F.J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All numerical results presented in this article are available upon request: inquiries should be directed to the authors.

Acknowledgments: J.D.P. wishes to acknowledge Giorgio Fasano for long-standing collaboration, for exchanging ideas on some of the topics discussed here, and for his constructive comments on the manuscript. J.D.P. also thanks AMPL Optimization and Wolfram Research for collaboration and technical support. The authors thank the reviewers for their constructive remarks and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Aste, T.; Weaire, D. *The Pursuit of Perfect Packing*, 2nd ed.; Taylor & Francis Group: Boca Raton, FL, USA, 2019. Available online: https://www.taylorfrancis.com/books/mono/10.1201/9781420068184/pursuit-perfect-packing-tomaso-aste-denis-weaire (accessed on 15 October 2024).
2. Friedman, E. Eric's Packing Center. 2024. Available online: https://erich-friedman.github.io/packing/index.html (accessed on 14 June 2024).
3. Specht, E. Packomania. 2024. Available online: http://www.packomania.com/ (accessed on 14 June 2024).
4. Wikipedia. Packing Problems. 2024. Available online: https://en.wikipedia.org/wiki/Packing_problems (accessed on 19 June 2024).
5. Castillo, I.; Kampas, F.J.; Pintér, J.D. Solving circle packing problems by global optimization: Numerical results and industrial applications. *Eur. J. Oper. Res.* **2008**, *191*, 786–802. [CrossRef]
6. Alvarez-Valdés, R.; Carravilla, M.A.; Oliveira, J.F. Cutting and packing. In *Handbook of Heuristics*; Martí, R., Pardalos, P.M., Resende, M.G., Eds.; Springer International Publishing: Berlin/Heidelberg, Germany, 2018; pp. 931–977. [CrossRef]

7. Bennell, J.; Oliveira, J.F. A tutorial in irregular shape packing problems. *J. Oper. Res. Soc.* **2009**, *60*, S93–S105. [CrossRef]
8. Scheithauer, G. *Introduction to Cutting and Packing Optimization. Problems, Modeling Approaches, Solution Methods*; Springer International Publishing AG: Cham, Switzerland, 2018. [CrossRef]
9. Fasano, G.; Pintér, J.D. (Eds.) *Optimized Packings with Applications*; Springer Nature: Cham, Switzerland, 2015. [CrossRef]
10. Dowsland, K.A.; Dowsland, W.B.; Bennell, J.A. Jostling for position: Local improvement for irregular cutting patterns. *J. Oper. Res. Soc.* **1998**, *49*, 647–658. [CrossRef]
11. Bennell, J.; Oliveira, J.F. The geometry of nesting problems: A tutorial. *Eur. J. Oper. Res.* **2008**, *184*, 397–415. [CrossRef]
12. Fasano, G. A global optimization point of view to handle non-standard object packing problems. *J. Glob. Optim.* **2013**, *55*, 279–299. [CrossRef]
13. Fasano, G. *Solving Non-Standard Packing Problems by Global Optimization and Heuristics*; SpringerBriefs in Optimization; Springer: Cham, Switzerland; Berlin/Heidelberg, Germany; New York, NY, USA; Dordrecht, The Netherlands; London, UK, 2014. [CrossRef]
14. Stoyan, Y.; Pankratov, A.; Romanova, T.; Fasano, G.; Pintér, J.D.; Stoian, Y.E.; Chugay, A. Optimized packings in space engineering applications: Part I. In *Modeling and Optimization in Space Engineering*; Fasano, G., Pintér, J.D., Eds.; Springer Optimization and Its Applications 144; Springer Nature: Cham, Switzerland, 2019. [CrossRef]
15. Stoyan, Y.; Grebennik, I.; Romanova, T.; Kovalenko, A. Optimized packings in space engineering applications: Part II. In *Modeling and Optimization in Space Engineering*; Fasano, G., Pintér, J.D., Eds.; Springer Optimization and Its Applications 144; Springer Nature: Cham, Switzerland, 2019. [CrossRef]
16. Stoyan, Y.; Yaskov, G.; Romanova, T.; Litvinchev, I.; Velarde Cantú, J.M.; Acosta, M.L. Packing spheres into a minimum-height parabolic container. *Axioms* **2024**, *13*, 396. [CrossRef]
17. Leao, A.A.S.; Toledo, F.M.B.; Oliveira, J.F.; Carravilla, M.A.; Alvarez-Valdés, R. Irregular packing problems: A review of mathematical models. *Eur. J. Oper. Res.* **2020**, *282*, 803–822. [CrossRef]
18. Oh, Y.; Witherell, P.; Luc, Y.; Sprock, T. Nesting and scheduling problems for additive manufacturing: A taxonomy and review. *Addit. Manuf.* **2020**, *36*, 101492. [CrossRef]
19. Horst, R.; Pardalos, P.M. (Eds.) *Handbook of Global Optimization*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1995; Volume 1. [CrossRef]
20. Pardalos, P.M.; Romeijn, H.E. (Eds.) *Handbook of Global Optimization*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2002; Volume 2. [CrossRef]
21. Pintér, J.D. Global optimization: Software, test problems, and applications. In *Handbook of Global Optimization*; Pardalos, P.M., Romeijn, H.E., Eds.; Kluwer Academic Publishers: Dordrecht, The Netherlands, 2002; Volume 2, pp. 515–569. [CrossRef]
22. Pintér, J.D. Software development for global optimization. In *Global Optimization: Methods and Applications*; Pardalos, P.M., Coleman, T.F., Eds.; Fields Institute Communications; American Mathematical Society: Providence, RI, USA, 2009; Volume 55, pp. 183–204. [CrossRef]
23. Pintér, J.D. How difficult is nonlinear optimization? A practical solver tuning approach, with illustrative results. *Ann. Oper. Res.* **2018**, *265*, 119–141. [CrossRef]
24. Floudas, C.A.; Pardalos, P.M.; Adjiman, C.S.; Esposito, W.R.; Gümüş, Z.H.; Harding, S.T.; Klepeis, J.L.; Meyer, C.A.; Schweiger, C.A. *Handbook of Test Problems in Local and Global Optimization*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1999. [CrossRef]
25. Dowsland, K.A.; Dowsland, W.B. Packing problems. *Eur. J. Oper. Res.* **1992**, *56*, 2–14. [CrossRef]
26. Bennell, J.; Scheithauer, G.; Stoyan, Y.; Romanova, T. Tools of mathematical modeling of arbitrary object packing problems. *Ann. Oper. Res.* **2010**, *179*, 343–368. [CrossRef]
27. Jones, D.R. A fully general, exact algorithm for nesting irregular shapes. *J. Glob. Optim.* **2014**, *59*, 367–404. [CrossRef]
28. Pankratov, A.; Romanova, T.; Litvinchev, I. Packing oblique 3D objects. *Mathematics* **2020**, *8*, 1130. [CrossRef]
29. Martí, R.; Pardalos, P.M.; Resende, M.G.C. (Eds.) *Handbook of Heuristics*; Springer Nature: Cham, Switzerland, 2018. [CrossRef]
30. Gendreau, M.; Potvin, J.-Y. (Eds.) *Handbook of Metaheuristics*, 3rd ed.; Springer Nature: Cham, Switzerland, 2019. [CrossRef]
31. Taillard, É.D. *Design of Heuristic Algorithms for Hard Optimization*; Springer Nature: Cham, Switzerland, 2023. [CrossRef]
32. Szabó, P.G.; Markót, M.C.; Csendes, T.; Specht, E.; Casado, L.G.; Garcia, I. *New Approaches to Circle Packing in a Square with Program Codes*; Springer Science + Business Media: New York, NY, USA, 2007. [CrossRef]
33. George, J.A.; George, J.M.; Lamar, B.W. Packing different-sized circles into a rectangular container. *Eur. J. Oper. Res.* **1995**, *84*, 693–712. [CrossRef]
34. Pintér, J.D.; Kampas, F.J. Nonlinear optimization in *Mathematica* with *MathOptimizer Professional*. *Math. Educ. Res.* **2005**, *10*, 1–18.
35. Huang, W.Q.; Li, Y.; Li, C.M.; Xu, R.C. New heuristics for packing unequal circles into a circular container. *Comput. Oper. Res.* **2006**, *33*, 2125–2142. [CrossRef]
36. Addis, B.; Locatelli, M.; Schoen, F. Efficiently packing unequal disks in a circle. *Oper. Res. Lett.* **2008**, *36*, 37–42. [CrossRef]
37. Wales, D.J.; Doye, J.P.K. Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *J. Phys. Chem. A* **1997**, *101*, 5111–5116. [CrossRef]
38. Zimmermann, A. Al Zimmermann's Programming Contests: Circle Packing. 2005. Available online: http://www.recmath.org/contest/CirclePacking/index.php (accessed on 19 June 2024).

39. Al-Mudahka, I.; Hifi, M.; M'Hallah, R. Packing circles in the smallest circle: An adaptive hybrid algorithm. *J. Oper. Res. Soc.* **2011**, *62*, 1917–1930. [CrossRef]
40. Specht, E. A precise algorithm to detect voids in polydisperse circle packings. *Proc. R. Soc. A* **2015**, *471*, 20150421. [CrossRef]
41. Romanova, T.; Pankratov, O.; Litvinchev, I.; Stetsyuk, P.; Lykhovyd, O.; Marmolejo-Saucedo, J.A.; Vasant, P. Balanced circular packing problems with distance constraints. *Computation* **2022**, *10*, 113. [CrossRef]
42. Grosso, A.; Jamali, A.R.M.J.U.; Locatelli, M.; Schoen, F. Solving the problem of packing equal and unequal circles in a circular container. *J. Glob. Optim.* **2010**, *47*, 63–81. [CrossRef]
43. Lü, Z.; Huang, W. PERM for solving circle packing problem. *Comput. Oper. Res.* **2008**, *35*, 1742–1755. [CrossRef]
44. Hifi, M.; M'Hallah, R. A literature review on circle and sphere packing problems: Models and methodologies. *Adv. Oper. Res.* **2009**, *2009*, 150624. [CrossRef]
45. López, C.O.; Beasley, J.E. A formulation space search heuristic for packing unequal circles in a fixed size circular container. *Eur. J. Oper. Res.* **2016**, *251*, 64–73. [CrossRef]
46. Ryu, J.; Lee, M.; Kim, D.; Kallrath, J.; Sugikara, K.; Kim, D.-S. VOROPACK-D: Real-time disk packing algorithm using Voronoi diagram. *Appl. Math. Comput.* **2020**, *375*, 125076. [CrossRef]
47. He, K.; Tole, K.; Ni, F.; Yuan, Y.; Liao, L. Adaptive large neighborhood search for solving the circle bin packing problem. *Comput. Oper. Res.* **2021**, *127*, 105140. [CrossRef]
48. AMPL Optimization. AMPL. 2024. Available online: https://ampl.com/ (accessed on 15 October 2024).
49. Wolfram Research. *Mathematica (Version 14.0)*; Wolfram Research, Inc.: Champaign, IL, USA, 2024. Available online: https://www.wolfram.com/mathematica/ (accessed on 15 October 2024).
50. Pintér, J.D. *Global Optimization in Action*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1996. [CrossRef]
51. Pintér, J.D. LGO—A program system for continuous and Lipschitz global optimization. In *Developments in Global Optimization*; Bomze, I.M., Csendes, T., Horst, R., Pardalos, P.M., Eds.; Springer: Boston, MA, USA, 1997. [CrossRef]
52. Pintér, J.D.; AMPL Optimization. AMPL-LGO Solver Engine. 2014. Available online: https://ampl.com/products/solvers/solvers-we-sell/lgo/ (accessed on 15 October 2024).
53. Pintér, J.D.; Kampas, F.J. MathOptimizer Professional for Mathematica. 2005. Available online: https://www.wolfram.com/products/applications/mathoptpro/ (accessed on 15 October 2024).
54. Wäscher, G.; Haußner, H.; Schumann, H. An improved typology of cutting and packing problems. *Eur. J. Oper. Res.* **2007**, *183*, 1109–1130. [CrossRef]
55. Pintér, J.D.; Kampas, F.J.; Castillo, I. Globally optimized packings of non-uniform size spheres in R^d: A computational study. *Optim. Lett.* **2018**, *12*, 585–613. [CrossRef]
56. Kampas, F.J.; Pintér, J.D.; Castillo, I. Optimal packing of general ellipses in a circle. In *Modeling and Optimization: Theory and Applications (MOPTA 2016)*; Takáč, M., Terlaky, T., Eds.; Springer: New York, NY, USA, 2017; pp. 23–37. [CrossRef]
57. Kampas, F.J.; Castillo, I.; Pintér, J.D. Optimized ellipse packings in regular polygons. *Optim. Lett.* **2019**, *13*, 1583–1613. [CrossRef]
58. Kampas, F.J.; Pintér, J.D.; Castillo, I. Packing ovals in optimized regular polygons. *J. Glob. Optim.* **2020**, *77*, 175–196. [CrossRef]
59. Kampas, F.J.; Pintér, J.D.; Castillo, I. Model development and solver demonstrations using randomized test problems. *Oper. Res. Forum* **2023**, *4*, 13. [CrossRef]
60. Castillo, I.; Pintér, J.D.; Kampas, F.J. The boundary-to-boundary p-dispersion configuration problem with oval objects. *J. Oper. Res. Soc.* **2024**, 1–11. [CrossRef]
61. Duriagina, Z.; Pankratov, A.; Romanova, T.; Litvinchev, I.; Bennell, J.; Lemishka, I.; Maximov, S. Optimized packing titanium alloy powder particles. *Computation* **2023**, *11*, 22. [CrossRef]
62. IPOPT: COIN-OR Interior Point Optimizer. 2024. Available online: https://github.com/coin-or/Ipopt (accessed on 14 June 2024).
63. Wächter, A.; Biegler, L.T. On the implementation of an interior-point filter line-search algorithm for large- scale nonlinear programming. *Math. Program.* **2006**, *106*, 25–57. [CrossRef]
64. IPOPT Linked to Mathematica. 2024. Available online: https://reference.wolfram.com/language/IPOPTLink/guide/IPOPTLink.html (accessed on 15 October 2024).
65. Zabinsky, Z.B. *Stochastic Adaptive Search for Global Optimization*; Springer Science + Business Media: New York, NY, USA, 2003. [CrossRef]
66. Zhigljavsky, A.; Žilinskas, A. *Stochastic Global Optimization*; Springer Science + Business Media: New York, NY, USA, 2008. [CrossRef]
67. De Haan, L.; Ferreira, A. *Extreme Value Theory: An Introduction*; Springer Science + Business Media: New York, NY, USA, 2006. [CrossRef]