

Global Optimization for Combinatorial Geometry Problems Revisited in the Era of LLMs

Timo Berthold^{1,2}, Dominik Kamp³, Gioni Mexi³, Sebastian Pokutta^{2,3}, and
Imre Pólik⁴

¹ Fair Isaac Deutschland GmbH, Germany, timoberthold@fico.com

² Technische Universität Berlin, Institut für Mathematik, Germany

³ Zuse Institute Berlin, Germany

⁴ Fair Isaac Corporation, USA, imre@polik.net

Abstract. Recent progress in LLM-driven algorithm discovery, exemplified by DeepMind's AlphaEvolve, has produced new best-known solutions for a range of hard geometric and combinatorial problems. This raises a natural question: to what extent can modern off-the-shelf global optimization solvers match such results when the problems are formulated directly as nonlinear optimization problems (NLPs)?

We revisit a subset of problems from the AlphaEvolve benchmark suite and evaluate straightforward NLP formulations with two state-of-the-art solvers, the commercial FICO Xpress and the open-source SCIP. Without any solver modifications, both solvers reproduce, and in several cases improve upon, the best solutions previously reported in the literature, including the recent LLM-driven discoveries. Our results not only highlight the maturity of generic NLP technology and its ability to tackle nonlinear mathematical problems that were out of reach for general-purpose solvers only a decade ago, but also position global NLP solvers as powerful tools that may be exploited within LLM-driven algorithm discovery.

Keywords: nonlinear optimization · circle packing · hexagon packing · distance ratio

1 Introduction

The rapid progress in global optimization technology over the past decade has substantially expanded the range of nonlinear, nonconvex problems that can be solved to proven global optimality, or at least to very high-quality solutions with reliable dual bounds. State-of-the-art academic solvers like SCIP [20] and commercial solvers like FICO[®] Xpress [3] combine spatial branch-and-bound, automatic linearization and convexification, sophisticated presolving, and increasingly powerful primal heuristics [4,5,6], enabling them to solve instances that would have been considered computationally prohibitive only a few years ago.

At the same time, recent developments in algorithm design based on Large Language Models (LLMs) have drawn renewed attention to long-standing geometric and combinatorial problems that can be formulated as nonlinear optimization models. Specifically, DeepMind presented the AlphaEvolve framework [19,25], which uses LLM-generated code in an evolutionary search to produce high-quality solutions for an extensive set of mathematical problems, including variants of circle packing, hexagon packing, and minimum-distance configurations of points. These breakthrough results raise a natural question: to what extent can state-of-the-art global optimization solvers match, or even surpass, such automatically discovered algorithms on challenging nonlinear problems?

More broadly, the last few years have seen rapid progress in LLM-driven discovery workflows that couple generative models with structured search and automated evaluation or verification. FunSearch combines an LLM with evolutionary program search and task-specific evaluators, enabling improvements for several discrete mathematical and algorithmic problems [28]. Complementary advances include AlphaDev, which used learning-based search to rediscover and improve low-level algorithms such as sorting routines [24], and AlphaGeometry, which combines neural generation with symbolic reasoning to solve olympiad-level geometry problems without human demonstrations [33]. Related LLM+evolution approaches have also been used to design effective heuristics for combinatorial optimization [23]. Taken together, these advances motivate revisiting classical mathematical optimization as a competitive and reliable baseline on the same challenging benchmarks.

In this work we revisit three problems from the AlphaEvolve benchmark suite and study them through the lens of Nonlinear Programming (NLP). An *NLP* is an optimization problem minimizing a nonlinear objective function over a feasible set defined by nonlinear constraints on continuous variables:

$$\min\{f(x) \mid g_k(x) \leq 0, \forall k \in \mathcal{K}, \ell \leq x \leq u\}, \quad (1)$$

where the objective $f(x)$ and all constraint functions $g_k : \mathbb{R}^n \rightarrow \mathbb{R}$ are factorable and all variable bounds $\ell, u \in \mathbb{R} := \mathbb{R} \cup \{\pm\infty\}$. The set $\mathcal{K} := \{1, \dots, m\}$ indexes the constraints. The circle packing problem, the minimum-distance ratio problem and the hexagon packing problem admit compact NLP models. This makes them good showcases for the power of modern global optimization tools: these combinatorial problems can be very intuitively modelled and effectively solved in their most natural form, as nonlinear optimization instances. We show that combined with off-the-shelf global optimization technology, these straightforward NLP formulations not only reproduce the best solutions reported in [19,25], but in several cases produce significantly better solutions. Our goal in doing so is to illustrate the power of modern general-purpose nonlinear optimization solvers rather than to perform a head-to-head comparison of solvers.

Next to presenting simple formulations that work and identifying some key modeling decisions that drive performance, our main contributions are:

Stronger solutions with unmodified solvers. With off-the-shelf Xpress and SCIP we obtain solutions that match, and in multiple cases improve, previously reported best-known results. Some problems were solved with only one solver, some used both. In this paper we do not make a distinction of which solver found which solution. All the solutions that our solvers produced were verified using the validation code in the AlphaEvolve repository.

Lessons learned. We reflect on the respective strengths and limitations of LLM-based approaches and nonlinear optimization, discuss modeling insights, and how global optimization software has become a complementary, industry-ready tool for challenging combinatorial problems.

These insights are discussed in Section 5, while Sections 2–4 present one model each, together with the accompanying computational results. Due to lack of space we will not comment on the dual bounds and how they could be improved (such as stronger formulations or symmetry breaking), even though it is a key differentiator between ad-hoc heuristics and general optimization solvers.

2 Minimizing the Ratio of Maximum to Minimum Distance

The problem of minimizing the ratio between the maximum and minimum pairwise distances in a finite point set, called the *min-max ratio problem*, has its origins in classical extremal geometry. Bateman and Erdős [2] formulated the question in terms of determining, for a given number of points, the configuration of points in the plane with mutual distances at least one that minimizes the diameter of the set. This is equivalent to minimizing the ratio of largest to smallest distance when the minimum is normalized to one. Bateman and Erdős provided optimal solutions for $n \leq 7$ in two dimensions. David Cantrell, see, e.g., [16], has contributed many best-known solutions in both 2D and 3D, while Audet et al. formulated the problem explicitly as a nonlinear optimization model and found some improving configurations for $n \leq 30$, using derivative-free optimization techniques [1].

2.1 Optimization model

In a slightly more general form, for given positive integers n and d , the goal of the min-max ratio problem is to find n distinct points in d -dimensional Euclidean space \mathbb{R}^d such that the ratio between the maximum and the minimum pairwise distance is minimized. This problem seeks a configuration of points that is as evenly spaced as possible, minimizing clustering and maximizing spatial efficiency. Let $\mathcal{N} = \{1, 2, \dots, n\}$ denote the set of points and $\mathcal{D} = \{1, 2, \dots, d\}$ denote the set of dimensions. We define the following decision variables:

$x_{i,k} \in \mathbb{R}$: coordinate of point $i \in \mathcal{N}$ in dimension $k \in \mathcal{D}$
 $t_{\min} \in \mathbb{R}_+$: minimum squared distance between any two points

$t_{\max} \in \mathbb{R}_+$: maximum squared distance between any two points

$r \in \mathbb{R}_+$: squared ratio variable, defined as $r = \frac{t_{\max}}{t_{\min}}$

The first observation is that it is more convenient and efficient to work with the ratio of the squared distances and avoid the square roots. For any pair of points $i, j \in \mathcal{N}$, the squared Euclidean distance is given by:

$$d_{ij}^2 = \sum_{k \in \mathcal{D}} (x_{i,k} - x_{j,k})^2$$

A key insight is that this problem exhibits *scale-invariance*: the optimal objective value remains unchanged if all lengths are multiplied by the same constant factor. We can therefore normalize the problem by fixing either t_{\min} or t_{\max} without loss of generality. This simplifies the formulation by reducing the number of nonlinear terms, thereby benefiting the solution process. We present two equivalent formulations:

Circle Packing Formulation By fixing $t_{\min} = 1$, the problem becomes a variant of the uniform circle packing problem where points maintain a pairwise distance of at least 1, with the maximum distance to be minimized. The optimization problem can be formulated as:

$$\min t_{\max} \tag{2}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{D}} (x_{i,k} - x_{j,k})^2 \geq 1, \quad \text{for all } i, j \in \mathcal{N}, i < j \tag{3}$$

$$\sum_{k \in \mathcal{D}} (x_{i,k} - x_{j,k})^2 \leq t_{\max}, \quad \text{for all } i, j \in \mathcal{N}, i < j \tag{4}$$

$$t_{\max} \geq 1 \tag{5}$$

Constraints (3) ensure that all pairwise squared distances are at least⁵ 1, while constraints (4) bound them by t_{\max} . Minimizing t_{\max} is then equivalent to minimizing the ratio $r = \frac{t_{\max}}{1} = t_{\max}$.

Dual Formulation. Alternatively, fixing $t_{\max} = 1$ yields a dual perspective where we maximize the minimum distance subject to a unit bound on the maximum distance:

$$\max t_{\min} \tag{6}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{D}} (x_{i,k} - x_{j,k})^2 \geq t_{\min}, \quad \text{for all } i, j \in \mathcal{N}, i < j \tag{7}$$

$$\sum_{k \in \mathcal{D}} (x_{i,k} - x_{j,k})^2 \leq 1, \quad \text{for all } i, j \in \mathcal{N}, i < j \tag{8}$$

$$0 \leq t_{\min} \leq 1 \tag{9}$$

⁵ For any (locally) optimal solution some of the inequalities in (3) will be satisfied with equality, otherwise the solution could be trivially improved by scaling.

Maximizing t_{\min} is then also equivalent to minimizing the ratio $r = \frac{1}{t_{\min}}$. Again, this simple flip renders a nonlinear rational objective into a linear objective, which is preferable for the solution process.

Both formulations yield nonconvex quadratically constrained optimization problems (QCPs) that can be solved efficiently by modern global optimization solvers. In our experiments the first model performed slightly better, but the difference was not very significant.

The power of the mathematical modeling approach is that we can handle point configurations in any dimension with the same model. Crucially, and that may be one of the reasons why optimization solvers perform great on this problem, is that it is essentially unconstrained (at least in its original form): any finite set of points has a largest and smallest pairwise distance, so their ratio is well-defined. There is no extra constraint that the point set has to satisfy (such as inclusion in a bounding region). The constraints get introduced only through the auxiliary variable for the objective.

2.2 Computational Results

We implemented the above model in the Mosel modeling language [14]. The resulting instances range in size from 7 variables and 10 constraints (three points in two dimensions) to 91 variables and 876 constraints (30 points in three dimensions). The improving solutions (rounded up to five decimal places) reported in Table 1 were obtained with this approach, and the corresponding configurations for the 2d case are illustrated in Figure 1. For many additional instances, the method reproduced the currently best-known solutions.

d	n	Squared distance ratio	Previous best	Source
2	16	12.88924	12.88927	[19,25]
2	21	17.77 499	17.776	[16]
2	22	19.05 398	19.055	[16]
2	29	25.92 460	25.929	[1]
3	14	4.165 78	4.16585	[19,25]

Table 1. Improving solutions for the min-max distance ratio problem in 2D and 3D

3 Packing Circles inside a Square or a Rectangle

The study of packing circles in polygonal regions dates back almost 200 years [9]. One of the most studied variants concerns packing a fixed number of unit circles into the smallest possible square; a good survey is given by Peikert [26]. For most instances of up to twenty circles, exact optimal packings are known and can be

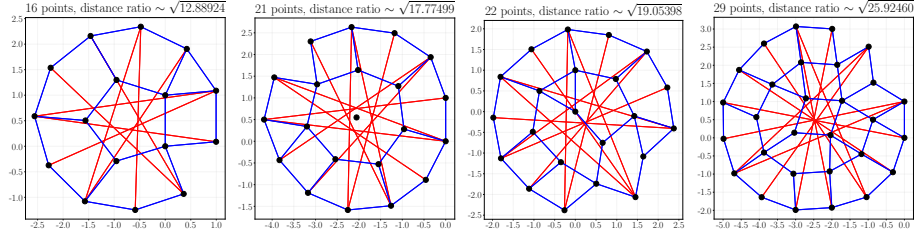


Fig. 1. Graphical representation of the solutions for the distance ratio problem in 2D. Red lines indicate maximum distance pairs, blue lines indicate minimum distance pairs.

derived using algebraic geometry techniques based on polynomial systems and Gröbner basis computations, see, e.g., [11].

In this work, we investigate a variant for which it is harder to prove optimality since both the positions and radii of the circles are decision variables. Specifically, we consider packing circles of variable radii into a unit square, or, in a mild relaxation, into a rectangle of perimeter four, while maximizing the sum of their radii. Best-known solutions are mostly due to Cantrell [17], but optimality proofs are only known for some trivial cases.

3.1 Optimization model

Given a positive integer n , we consider the problem of packing n circles inside a rectangle such that the sum of their radii is maximized. We examine two variants: packing circles into any rectangle of perimeter 4, and the special case of packing circles into a unit square (which has perimeter 4). A key advantage of mathematical optimization approaches is that the optimization model can be changed easily. This facilitates exploring related problems by simply adding or modifying constraints, such as for the two problem variants shown below.

Let $\mathcal{N} = \{1, 2, \dots, n\}$ denote the set of circles. We define the following decision variables:

$(x_i, y_i) \in \mathbb{R}^2$: coordinates of the center of circle $i \in \mathcal{N}$

$r_i \in \mathbb{R}_+$: radius of circle $i \in \mathcal{N}$

$\alpha \in \mathbb{R}_+$: width of the rectangle

We consider the problem of packing circles in a rectangle with fixed perimeter $P = 4$ and variable side lengths. We introduce a decision variable α representing the width and set the height to $H = \frac{P}{2} - \alpha = 2 - \alpha$. We can assume without loss of generality that $\alpha \leq 1$ (i.e., α is the shorter side). The optimization problem

can be formulated as:

$$\max \sum_{i \in \mathcal{N}} r_i \quad (10)$$

$$\text{s.t. } r_i \leq x_i \leq \alpha - r_i, \quad \text{for all } i \in \mathcal{N} \quad (11)$$

$$r_i \leq y_i \leq (2 - \alpha) - r_i, \quad \text{for all } i \in \mathcal{N} \quad (12)$$

$$(x_i - x_j)^2 + (y_i - y_j)^2 \geq (r_i + r_j)^2, \quad \text{for all } i, j \in \mathcal{N}, i < j \quad (13)$$

$$0 \leq r_i \leq \frac{\alpha}{2}, \quad \text{for all } i \in \mathcal{N} \quad (14)$$

$$0 < \alpha \leq 1 \quad (15)$$

The objective (10) is to maximize the sum of all radii. Note that this is fundamentally different from maximizing the total area covered by circles, which would be $\pi \sum_{i \in \mathcal{N}} r_i^2$, whereas the linear objective tends to favor more balanced distributions of circle sizes. Constraints (11) and (12) ensure that each circle remains entirely within the rectangle: the center coordinates must maintain a distance of at least r_i from all rectangle boundaries. Constraints (13) ensure that no two circles overlap by requiring that the Euclidean distance⁶ between any pair of circle centers is at least the sum of their radii. Constraints (14) provide an upper bound on each radius: no circle can have a diameter exceeding the width of the rectangle α (which is the shorter side by assumption). Finally, constraint (15) bounds the width variable.

The aspect ratio determined by α is a decision variable that can be modified to maximize the sum of radii for a given number of circles. We can trivially change this formulation to packing into a unit square by fixing $\alpha = 1$. This is a crucial property of mathematical optimization modeling: the user needs to change only the model and does not have to worry about whether this changes the algorithm: the solvers will take care of that. This contrasts with many heuristic approaches, including LLM-generated ones, in which a new set of heuristics often needs to be developed once the model formulation changes.

Again, we have only linear and nonconvex quadratic constraints.

3.2 Computational Results

First, let us focus on the restricted problem, packing circles into a square. The following improving solution (floored to five decimal digits) has been found⁷ with both solvers (see Table 2). The solution is visualized in Figure 2.

Now we can turn to the slightly relaxed version, where instead of a square we are trying to pack into a rectangle of perimeter 4. Obviously, all the previous solutions are still feasible for this relaxed problem, but we can do slightly better.

⁶ Similar to the distance ratio problem, it is again advantageous to work with squared distances.

⁷ We have found possibly even more improving solutions, but the original source [17] for the best-known packings lists only three digits after the decimal. In almost all cases, we managed to match the best-known solutions from literature.

Variant	n	Sum of radii	Previous best	Source
square	32	2.93957	2.93794	[19,25]
rectangle	26	2.63930	2.638	[16]
rectangle	27	2.69015	2.687	[16]

Table 2. Improving solutions for the circle packing problem

We found the following improving solutions, again floored to five decimal digits (see Table 2). The solutions are visualized in Figure 2. Not surprisingly, the solutions all tend to use a rectangle that is almost square.

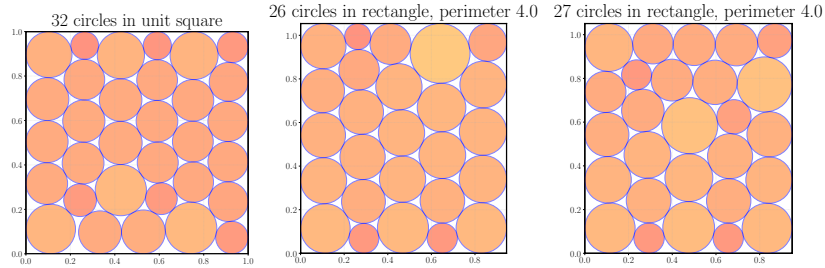


Fig. 2. Graphical representation of the solutions for the circle packing problem. Left: square variant ($n=32$). Middle and right: rectangle variant ($n=26$, $n=27$).

4 Packing Unit Regular Hexagons inside a Regular Hexagon

Packing various shapes into other shapes in an optimal way has a vast literature, from classical geometry problems and puzzles [8,15], to packing problems solved for designing packaging and packing boxes into shipping containers [10,35]. In general, the more complicated the shape to be packed is, the harder the problems become both to analyze theoretically and to solve practically. Problems with unit circles are well-studied (compare Section 3), but more complex shapes are often only handled with ad-hoc heuristics.

In this section we consider the problem of packing n regular hexagons with unit side length into a regular hexagon of minimum side length R . Each inner hexagon can be positioned and rotated freely. This problem is significantly more involved than circle packing due to the possible rotation of the non-circular geometry of hexagons. However, it is a special case of the polygon packing problem, where two-dimensional geometrical objects bounded by an arbitrary closed

sequence of non-intersecting line segments must be packed into a container in some efficient way. An optimization model for the polygon packing problem with a fixed size container was developed by [22], which was later generalized to convex objects by [32] and afterwards even further to also allow nonconvex objects by [27]. These approaches construct so-called quasi-phi-functions for certain object geometries to formulate pairwise non-overlapping conditions among them. The following hexagon packing formulation adopts this approach in a simplified way by defining Farkas multipliers (see, e.g., [7]) on pairwise hexagon intersections in order to exclude interior overlaps, respectively.

4.1 Optimization model

Let $\mathcal{N} = \{1, 2, \dots, n\}$ denote the set of inner hexagons and $\mathcal{S} = \{0, 1, 2, 3, 4, 5\}$ denote the six sides of any hexagon. We use the following geometric constants:

$\rho = \frac{\sqrt{3}}{2}$: inradius (distance from center to edge, also called apothem)
 $\phi = \frac{\pi}{3}$: angular separation between adjacent vertices

We define the following decision variables:

$R \in \mathbb{R}_+$: side length of the outer hexagon (to be minimized)
 $(x_i, y_i) \in \mathbb{R}^2$: coordinates of the center of inner hexagon $i \in \mathcal{N}$
 $\theta_i \in [0, \phi]$: rotation angle of inner hexagon $i \in \mathcal{N}$

To facilitate the formulation of the geometric constraints we introduce:

$a_{i,j}, b_{i,j} \in \mathbb{R}$: normal vector components for side $j \in \mathcal{S}$ of hexagon $i \in \mathcal{N}$
 $c_{i,j} \in \mathbb{R}$: offset term for side $j \in \mathcal{S}$ of hexagon $i \in \mathcal{N}$
 $\lambda_{i,j,k} \in \mathbb{R}_+$: Farkas multipliers for separation of hexagons $i, j \in \mathcal{N}$ with $i < j$,
 where $k \in \{1, \dots, 12\}$

The optimization problem can be formulated as:

$$\min R \tag{16}$$

$$\begin{aligned} \text{s.t. } & R\rho + \sin(k\phi)(x_i + \sin(\theta_i + (j + 0.5)\phi)) \\ & + \cos(k\phi)(y_i + \cos(\theta_i + (j + 0.5)\phi)) \geq 0, \quad \forall i \in \mathcal{N}, j, k \in \mathcal{S} \end{aligned} \tag{17}$$

$$a_{i,j} = \sin(\theta_i + j\phi), \quad \forall i \in \mathcal{N}, j \in \mathcal{S} \tag{18}$$

$$b_{i,j} = \cos(\theta_i + j\phi), \quad \forall i \in \mathcal{N}, j \in \mathcal{S} \tag{19}$$

$$c_{i,j} = a_{i,j}x_i + b_{i,j}y_i - \rho, \quad \forall i \in \mathcal{N}, j \in \mathcal{S} \tag{20}$$

$$\sum_{k=1}^{12} \lambda_{i,j,k} = 1, \quad \forall i, j \in \mathcal{N}, i < j \tag{21}$$

$$\sum_{k=0}^5 \lambda_{i,j,k+1} a_{i,k} + \sum_{k=0}^5 \lambda_{i,j,k+7} a_{j,k} = 0, \quad \forall i, j \in \mathcal{N}, i < j \tag{22}$$

$$\sum_{k=0}^5 \lambda_{i,j,k+1} b_{i,k} + \sum_{k=0}^5 \lambda_{i,j,k+7} b_{j,k} = 0, \quad \forall i, j \in \mathcal{N}, \quad i < j \quad (23)$$

$$\sum_{k=0}^5 \lambda_{i,j,k+1} c_{i,k} + \sum_{k=0}^5 \lambda_{i,j,k+7} c_{j,k} \geq 0, \quad \forall i, j \in \mathcal{N}, \quad i < j \quad (24)$$

$$0 \leq \theta_i \leq \phi, \quad \forall i \in \mathcal{N} \quad (25)$$

$$\lambda_{i,j,k} \geq 0, \quad \forall i, j, k \quad (26)$$

$$R \geq \sqrt{n} \quad (27)$$

Constraints (17) ensure that all vertices of each inner hexagon lie within the outer hexagon. For a regular hexagon centered at (x_i, y_i) with rotation angle θ_i , the vertices are located at positions $(x_i + \sin(\theta_i + (j + 0.5)\phi), y_i + \cos(\theta_i + (j + 0.5)\phi))$ for $j \in \mathcal{S}$. Each vertex must satisfy all six half-space constraints defining the outer hexagon.

Constraints (18)–(20) define the half-space representation of the inner hexagons. Each side $j \in \mathcal{S}$ of hexagon $i \in \mathcal{N}$ is characterized by a normal vector $(a_{i,j}, b_{i,j})$ and an offset $c_{i,j}$, representing the inequality $a_{i,j}x + b_{i,j}y \geq c_{i,j}$.

Constraints (21)–(24) represent Farkas-based separation conditions to ensure that no two inner hexagons overlap. These constraints are based on the Farkas lemma for certifying infeasibility of sets of linear constraints. For each pair of hexagons (i, j) with $i < j$, we introduce 12 Farkas multipliers $\lambda_{i,j,k}$ (one for each side of the two hexagons). The constraints ensure that the mutual system of inequalities defining the intersection of both hexagons is degenerate, i.e., no point can simultaneously lie in the interior of both hexagons, thereby guaranteeing non-overlapping configurations.

Constraint (25) restricts the rotation angle to $[0, \phi]$ due to the six-fold rotational symmetry of regular hexagons because rotations beyond ϕ are equivalent to rotations within this range. Finally, the lower bound on R (27) is implied by elementary geometry. The area of the outer hexagon must be at least the total area of the n contained unit hexagons.

Due to the trigonometric functions, these problems are general nonconvex nonlinear. Even worse, there are nonlinear equality constraints in the formulation we presented. This generally can pose a challenge to any solution approach, as equations are naturally harder to satisfy. Global solvers handle this challenge in various ways, (both Xpress and SCIP use an outer approximation method). At the same time, the stronger constrainedness makes this problem also more challenging to create ad-hoc heuristics for. This is the area where we can report the largest improvements over the previous best solutions, indicating that modern global solvers are well positioned for this strongly constrained problem – a situation that we are used to from mixed-integer linear optimization for years.

The solution approach can easily be adapted to pack other regular polygons (triangles, squares, pentagons, etc.) by just changing n , recomputing ρ, ϕ , and rerunning the solver. Actually, any convex polyhedral objects can be handled in a structurally similar way even in higher dimensional domains as long as the

linear outer formulations are known because the non-overlapping constraints solely rely on the definition of feasible Farkas multipliers for the intersections. This is one example that demonstrates the generalizable nature of mathematical optimization.

4.2 Computational Results

The following solutions were produced (often within minutes) with a run using only default settings. The solvers managed to match the best-known solutions for up to 10 hexagons. After that, we found solutions that were better than the best-known solutions, except for 13 hexagons, where the known trivial (and most likely, optimal) solution with side length 4 was reproduced. In particular, the following improving solutions have been found, ceiled to five decimal digits (see Table 3 and Figure 3).

n	Side of containing hexagon	Previous best	Source
11	3.92485	3.93010	[19,25]
12	3.94165	3.94192	[19,25]
14	4.26900	4.27240	[18]
15	4.44769	4.45406	[18]
16	4.52788	4.53633	[18]

Table 3. Improving solutions for the hexagon packing problem

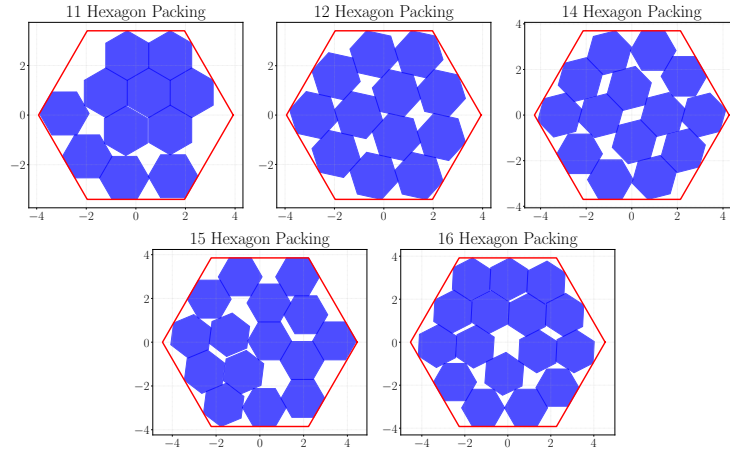


Fig. 3. Graphical representation of the solutions for the hexagon packing problem

5 Conclusion

Code-generating LLMs provide an effective entry point into new problem domains and ideally facilitate a technology-agnostic exploration of solution approaches. In the context of the present work, they also helped to draw attention to classical problems where further improvements are still possible and that had not been revisited by mathematical optimization approaches for some time. Our results show that off-the-shelf global optimization software can solve straightforward formulations of hard combinatorial geometry problems within seconds or minutes. This highlights the substantial progress achieved in recent years and suggests that nonlinear optimization is approaching the role that (mixed-integer) linear optimization has played for at least two decades: a robust, performant, readily available black-box technology.

At the same time, mathematical optimization can meet several expectations often associated with LLM-based approaches. Combinatorial and other mathematical problems can be formulated in a very natural way, and changes in the problem definition typically require only minor modifications to the model and are thereby quickly realized. Solutions can then be obtained quickly, presumably considerably faster than generating, refining, and testing solver-specific code through dozens of LLM-driven code-generation iterations.

Nevertheless, the two technologies should not be viewed as competing, but rather as complementary. The solution-method-agnostic prototyping capabilities of an LLM-based code generation are a powerful tool for exploring new problems and formulating new solution approaches. Curiously, in a study with OpenEvolve [31] on a problem of packing 26 circles specifically, the LLM converged to using an optimization approach [30], namely a Sequential Least Squares Programming formulation with a local solver implemented in SciPy [29,34]. Furthermore, the concept of using LLMs as a support tool for the authoring of optimization models, as, e.g., in [12,13,21] is a very promising one.

It is noteworthy that in our experiments the smallest improvements were observed for the relatively unconstrained min-max ratio problem, whereas the largest gains were achieved for the highly constrained hexagon packing problem. Although based on a limited set of examples, this suggests that optimization-based approaches become increasingly competitive the more constrained the problems are. Since real-world industrial problems are typically characterized by large and diverse constraint sets, these observations further support the view that global optimization has matured into an industry-ready technology.

Acknowledgments. We are grateful for the contributions of Liding Xu (ZIB) and the FICO Xpress team, in particular Bruno Vieira and Susanne Heipcke, for their contributions during an early phase of this project.

Research reported in this paper was partially supported through the Research Campus Modal funded by the German Federal Ministry of Education and Research (fund numbers 05M14ZAM, 05M20ZBM) and the Deutsche Forschungsgemeinschaft (DFG) through the DFG Cluster of Excellence MATH+.

References

1. Audet, C., Fournier, X., Hansen, P., Messine, F.: A note on diameters of point sets. *Optimization Letters* **4**(4), 585–595 (2010)
2. Bateman, P., Erdős, P.: Geometrical extrema suggested by a lemma of Besicovitch. *The American Mathematical Monthly* **58**(5), 306–314 (1951)
3. Belotti, P., Berthold, T., Gally, T., Gottwald, L., Pólik, I.: Solving MINLPs to global optimality with FICO Xpress Global. *Optimization* pp. 1–19 (2025). <https://doi.org/10.1080/02331934.2025.2595437>
4. Berthold, T.: Heuristic algorithms in global MINLP solvers. Ph.D. thesis, Technische Universität Berlin (2014)
5. Berthold, T.: A computational study of primal heuristics inside an MI(NL)P solver. *Journal of Global Optimization* **70**(1), 189–206 (2018). <https://doi.org/10.1007/s10898-017-0600-3>
6. Berthold, T., Lodi, A., Salvagnin, D.: *Primal Heuristics in Integer Programming*. Cambridge University Press (2025)
7. Bertsimas, D., Tsitsiklis, J.N.: *Introduction to linear optimization*, vol. 6. Athena scientific Belmont, MA (1997)
8. Bezdek, K.: *Classical topics in discrete geometry*. Springer Science & Business Media (2010)
9. Bolyai, W.: *Tentamen Juventutem Studiosam in Elementa Matheseos Purae, Elementaris Ac Sublimioris. Typis Collegii Reformatorem, Marosvásárhely* (1832), two volumes
10. Bortfeldt, A., Wäscher, G.: Constraints in container loading—a state-of-the-art review. *European Journal of Operational Research* **229**(1), 1–20 (2013)
11. Buchberger, B.: A theoretical basis for the reduction of polynomials to canonical forms. *ACM SIGSAM Bulletin* **10**(3), 19–29 (1976)
12. Chen, H., Constante-Flores, G.E., Mantri, K.S.I., Kompalli, S.M., Ahluwalia, A.S., Li, C.: Optichat: Bridging optimization models and practitioners with large language models. *INFORMS Journal on Data Science* (2025)
13. Chen, Z., Zhang, X., Zope, H., Barbalho, H., Mellou, K., Molinaro, M., Kulkarni, J., Menache, I., Li, S.: OptiMind: Teaching LLMs to think like optimization experts. *arXiv preprint arXiv:2509.22979* (2025)
14. Colombani, Y., Heipcke, S.: Mosel: an extensible environment for modeling and programming solutions. In: *Proceedings of CP-AI-OR*. vol. 2, pp. 277–290 (2002)
15. Conway, J.H., Sloane, N.J.A.: *Sphere packings, lattices and groups*, vol. 290. Springer Science & Business Media (2013)
16. Friedman, E.: Packing center — circle packings and related problems. <https://erich-friedman.github.io/packing/>, online Resource
17. Friedman, E.: Circles in squares (2012), <https://erich-friedman.github.io/packing/cirRsqu/>
18. Friedman, E.: Hexagons in hexagons (2015), <https://erich-friedman.github.io/packing/hexinhex/>
19. Georgiev, B., Gómez-Serrano, J., Tao, T., Wagner, A.Z.: Mathematical exploration and discovery at scale. *arXiv preprint arXiv:2511.02864* (2025)
20. Hojny, C., Besançon, M., Bestuzheva, K., Borst, S., Chmiela, A., Dionísio, J., Eifler, L., Ghannam, M., Gleixner, A., Göß, A., Hoen, A., van der Hulst, R., Kamp, D., Koch, T., Kofler, K., Lentz, J., Maher, S.J., Mexi, G., Mühmer, E., Pfetsch, M.E., Pokutta, S., Serrano, F., Shinano, Y., Turner, M., Vigerske, S.,

- Walter, M., Weninger, D., Xu, L.: The SCIP Optimization Suite 10.0. Technical report, Optimization Online (2025), <https://optimization-online.org/2025/11/the-scip-optimization-suite-10-0/>
21. Huang, C., Tang, Z., Hu, S., Jiang, R., Zheng, X., Ge, D., Wang, B., Wang, Z.: Orlm: A customizable framework in training large models for automated optimization modeling. *Operations Research* (2025)
 22. Kallrath, J.: Cutting circles and polygons from area-minimizing rectangles. *Journal of Global Optimization* **43**(2), 299–328 (2009)
 23. Liu, F., Tong, X., Yuan, M., Lin, X., Luo, F., Wang, Z., Lu, Z., Zhang, Q.: Evolution of heuristics: Towards efficient automatic algorithm design using large language model. *arXiv preprint arXiv:2401.02051* (2024), <https://arxiv.org/abs/2401.02051>
 24. Mankowitz, D.J., Michi, A., Zhernov, A., Gelmi, M., Selvi, M., Paduraru, C., Leurent, E., Iqbal, S., Lespiau, J.B., Ahern, A., Köppe, T., Millikin, K., Gaffney, S., Elster, S., Broshear, J., Gamble, C., Milan, K., Tung, R., Hwang, M., Cemgil, T., Barekatin, M., Li, Y., Mandhane, A., Hubert, T., Schrittwieser, J., Hassabis, D., Kohli, P., Riedmiller, M., Vinyals, O., Silver, D.: Faster sorting algorithms discovered using deep reinforcement learning. *Nature* **618**(7964), 257–263 (2023). <https://doi.org/10.1038/s41586-023-06004-9>
 25. Novikov, A., Vü, N., Eisenberger, M., Dupont, E., Huang, P.S., Wagner, A.Z., Shirobokov, S., Kozlovskii, B., Ruiz, F.J., Mehrabian, A., et al.: Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131* (2025)
 26. Peikert, R., Würtz, D., Monagan, M., de Groot, C.: Packing circles in a square: a review and new results. In: *System Modelling and Optimization: Proceedings of the 15th IFIP Conference Zurich, Switzerland, September 2–6, 1991*. pp. 45–54. Springer (2007)
 27. Romanova, T., Bennell, J., Stoyan, Y., Pankratov, A.: Packing of concave polyhedra with continuous rotations using nonlinear optimisation. *European Journal of Operational Research* **268**(1), 37–53 (2018)
 28. Romera-Paredes, B., Barekatin, M., Novikov, A., Balog, M., Kumar, M.P., Dupont, E., Ruiz, F.J.R., Ellenberg, J.S., Wang, P., Fawzi, O., Kohli, P., Fawzi, A.: Mathematical discoveries from program search with large language models. *Nature* **625**(7995), 468–475 (2024). <https://doi.org/10.1038/s41586-023-06924-6>
 29. SciPy API: Optimization and root finding (2025), <https://docs.scipy.org/doc/scipy/reference/optimize.minimize-slsqp.html>
 30. Sharma, A.: Advanced circle packing for n=26 circles in a unit square (2025), https://github.com/algorithmicsuperintelligence/openevolve/blob/main/examples/circle_packing/best_program.py
 31. Sharma, A.: Openevolve: an open-source evolutionary coding agent (2025), <https://github.com/algorithmicsuperintelligence/openevolve>
 32. Stoyan, Y., Pankratov, A., Romanova, T.: Quasi-phi-functions and optimal packing of ellipses. *Journal of Global Optimization* **65**(2), 283–307 (2016)
 33. Trinh, T.H., Wu, Y., Le, Q.V., He, H., Luong, T.: Solving olympiad geometry without human demonstrations. *Nature* **625**(7995), 476–482 (2024). <https://doi.org/10.1038/s41586-023-06747-5>
 34. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al.: SciPy 1.0: fundamental algorithms for scientific computing in python. *Nature methods* **17**(3), 261–272 (2020)

35. Wäscher, G., Haufner, H., Schumann, H.: An improved typology of cutting and packing problems. *European journal of operational research* **183**(3), 1109–1130 (2007)