OpenEvolve示例：circle_packing

概述

对 OpenEvolve 项目中 circle_packing 示例的分析，这是经典的**圆填充优化问题**，具体来说：

问题定义

目标：在单位正方形（1×1）内放置 26 个互不重叠的圆，最大化所有圆的半径之和。

约束条件

1. **边界约束**：每个圆必须完全位于单位正方形内
2. **非重叠约束**：任意两个圆不能重叠
3. **数量约束**：必须恰好放置 26 个圆
4. **半径约束**：所有圆的半径必须为非负数

优化目标

最大化目标函数：Σr_i（所有圆的半径之和）

参考基准

根据 AlphaEvolve 论文，该问题的最优解约为 2.635。OpenEvolve 项目通过进化算法成功达到了 2.634，与基准结果相差仅 0.04%。

问题特点

1. **组合优化问题**：需要同时优化圆的位置和半径
2. **非线性约束**：非重叠约束是二次约束
3. **连续变量**：圆心坐标和半径都是连续变量
4. **高维搜索空间**：26 个圆 × 3 个变量（x, y, r）= 78 维搜索空间

复杂度分析

1. 离散版本的NP-hard性

- **决策问题**：给定n个圆和容器，判断是否能将所有圆放入容器而不重叠 → 这是NP-hard的
- 可以归约到**正方形打包问题（Square Packing）**，后者是已知的NP-hard问题
- 也可以归约到**圆装箱问题（Circle Bin Packing）**

2. 连续版本的复杂性

这个具体问题（最大化半径之和）是**连续优化版本**，但：

- 即使半径是连续变量，搜索空间是指数级的
- 需要同时优化位置（连续）和半径（连续）
- 非重叠约束是非凸的二次约束，使得问题非凸且高度非线性

3. 相关问题的复杂度

- **等圆填充（Equal Circle Packing）**：在容器内放置n个等半径的圆以最大化半径 → 已被证明是NP-hard
- **不等圆填充（Unequal Circle Packing）**：更一般的情况，复杂度更高
- **球体填充（Sphere Packing）**：3D版本也是NP-hard

4. 实际意义

正因为是NP-hard问题，所以：

- 没有已知的多项式时间精确算法
- 需要启发式方法（如进化算法、模拟退火、局部搜索）
- OpenEvolve使用进化算法是处理这类问题的合适方法

结论

这个圆填充优化问题是**NP-hard问题**，这也是为什么需要像OpenEvolve这样的进化算法来寻找高质量近似解，而不是期望在多项式时间内找到精确最优解。

在 OpenEvolve 中的实现

整个演化分两个阶段，分别侧重探索与挖掘：

阶段 1：初始探索

- 采用构造式策略，把圆放在关键位置
- 尝试多种几何模式（同心环、网格、六边形等）
- 加入简单优化例程，在不重叠前提下尽可能放大半径

关键配置：

代码块

```
max_iterations: 100
population_size: 60
num_islands: 4
exploitation_ratio: 0.7
```

阶段 2：突破平台

当解在 2.377 附近陷入平台后，调整参数以鼓励更激进的创新：

- 扩大种群规模，提高多样性
- 降低挖掘比例，倾向探索
- 更新系统提示词，引导尝试不同优化技术
- 允许生成更长、更复杂的代码

关键配置：

代码块

```
max_iterations: 100
population_size: 70
num_islands: 5
exploitation_ratio: 0.6
```

进化过程：

- 初始方案：简单的同心圆环排列
- 中间改进：六边形排列、网格排列
- 最终方案：使用 SLSQP 算法的约束优化方法

这是一个典型的**计算几何优化问题**，在材料科学、包装设计、无线网络布局等领域有实际应用。

项目	内容
任务	把 26 个圆无重叠地放进单位正方形，最大化半径之和
文献目标值	≈ 2.635（AlphaEvolve 论文）
OpenEvolve 结果	2.634（99.97 % 文献值）

两阶段演化流程

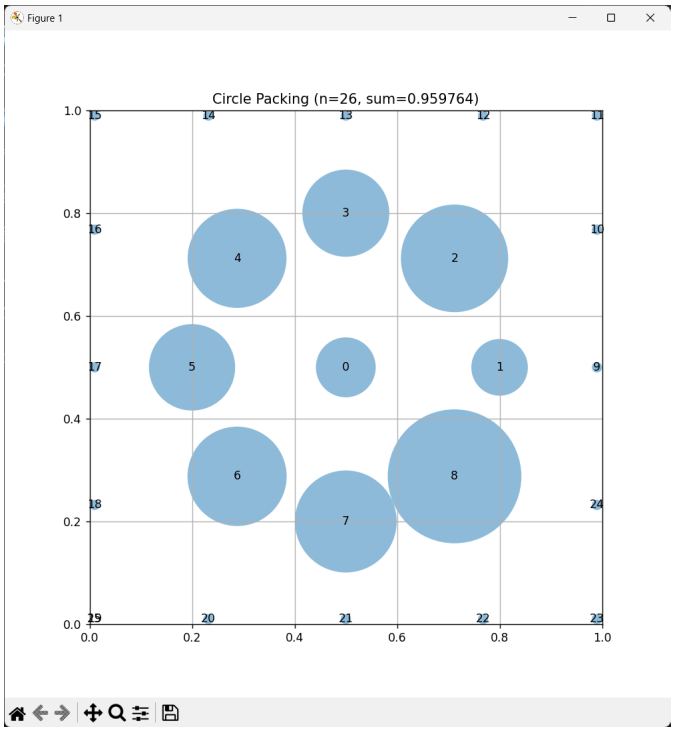
阶段	人口规模	岛屿数	利用比率	核心策略
① 探索期	60	4	0.7	多样构造：同心环、六边形、网格
② 突破期	70	5	0.6	鼓励激进创新，允许更长代码

算法跃迁快照

代数	关键特征	半径和
Gen 0	手工三环布局	0.96
Gen 10	六边形+微调	1.80
Gen 100	错位网格+变半径	2.20
终局	`scipy.optimize.minimize(SLSQP)` 约束优化	**2.634**

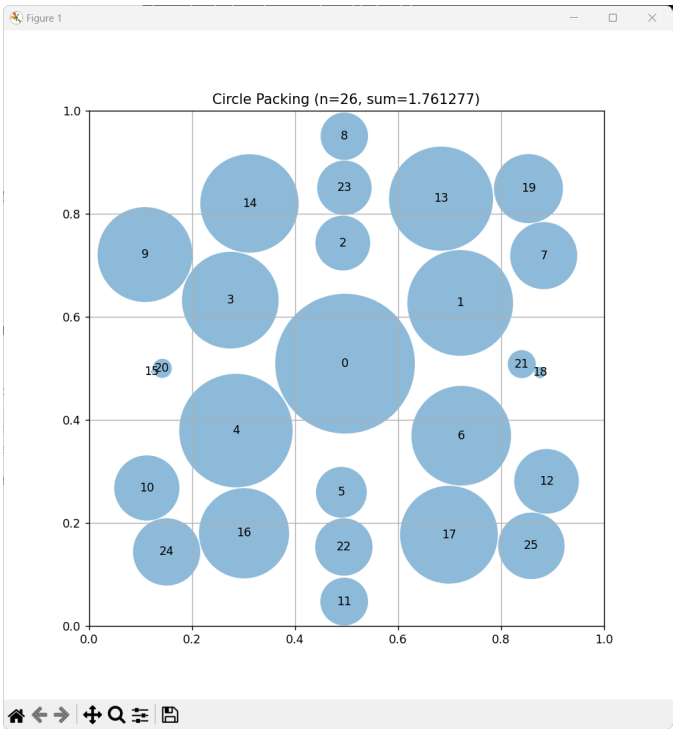
初始程序

Sum of radii: 0.9597642169962064



10次迭代

Sum of radii: 1.761277131744421

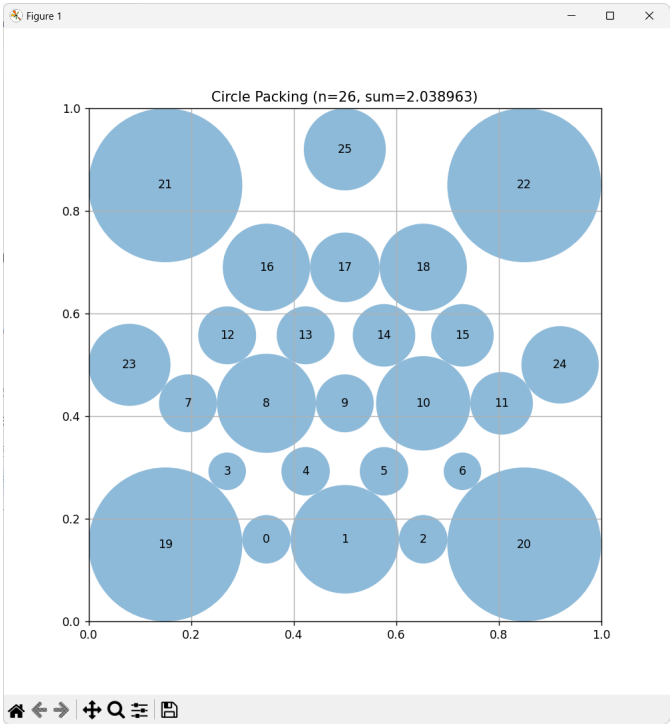


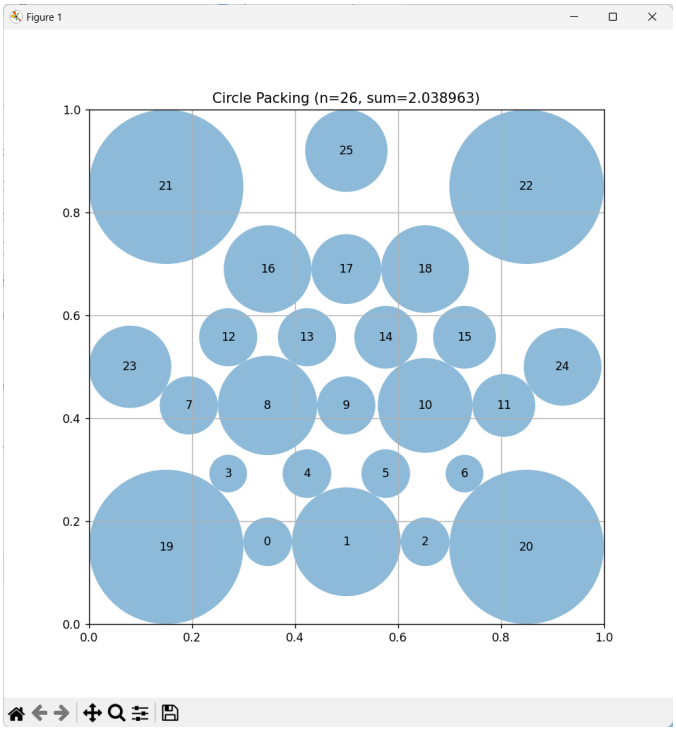
20次迭代

Sum of radii: 2.038963325338827

30次迭代

Sum of radii: 2.038963325338827



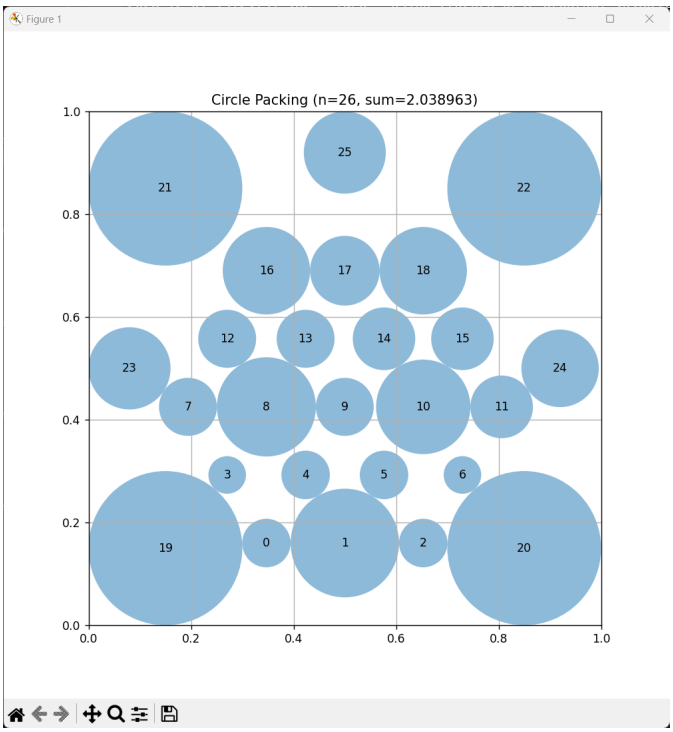


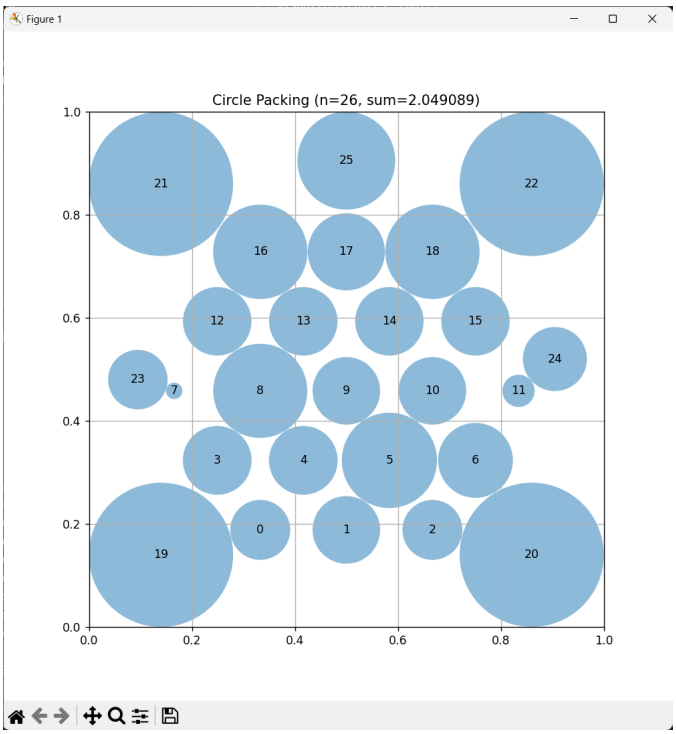
40次迭代

Sum of radii: 2.038963325338827

50次迭代

Sum of radii: 2.049089019852698



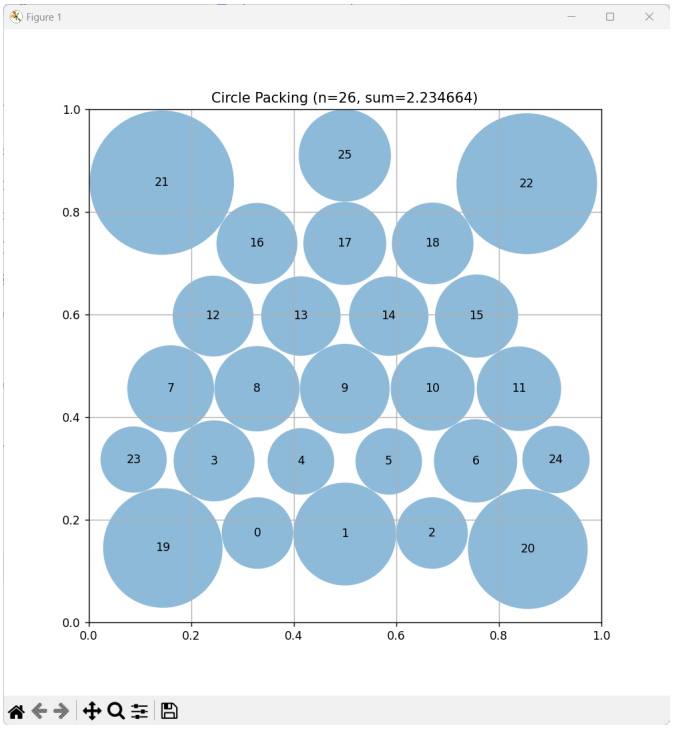


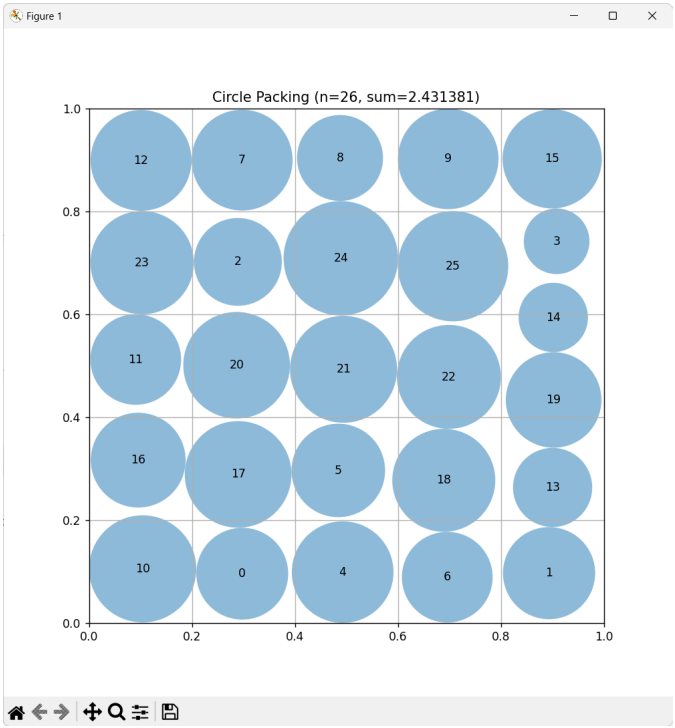
100次迭代

Sum of radii: 2.2346635179560645

200次迭代

Sum of radii: 2.4313805921199196





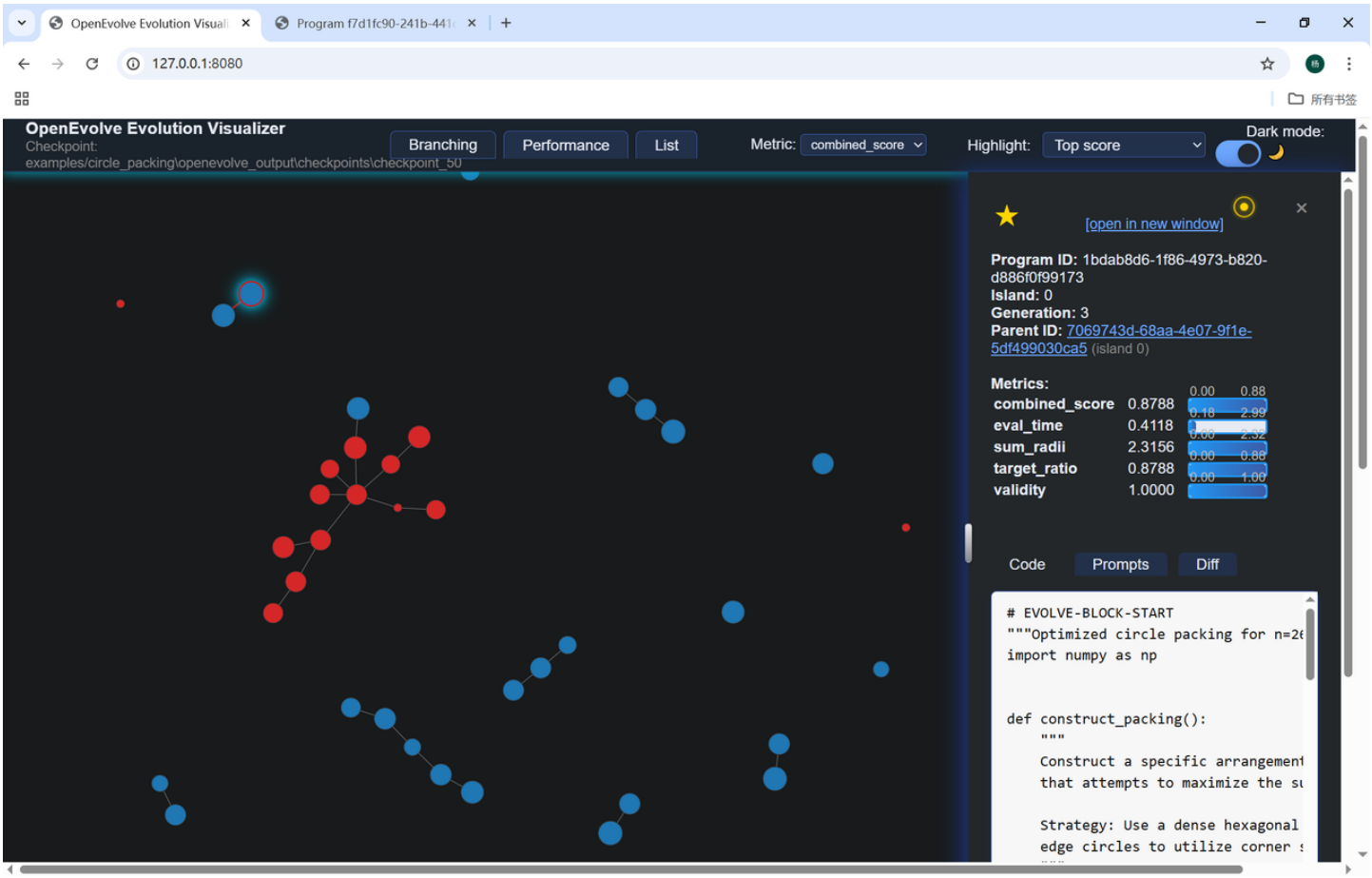
一句话总结

OpenEvolve 从“几何拼积木”自动进化到“数学规划”，在零人工提示下把半径和推到文献级精度，验证了其在数学优化问题上的算法发现能力。

代码块

```
# Phase 1: Initial exploration
python openevolve-run.py examples/circle_packing/initial_program.py examples/circle_packing/evaluator.py  --config examples/circle_packing/config_phase_1.yaml  --iterations 100

# Phase 2: Breaking through the plateau
python openevolve-run.py examples/circle_packing/openevolve_output/checkpoints/checkpoint_100/best_program.py  examples/circle_packing/evaluator.py --config examples/circle_packing/config_phase_2.yaml --iterations 100
```


OpenEvolve Evolution Visuali
Program f7d1fc90-241b-441
127.0.0.1:8080
所有书签
OpenEvolve Evolution Visualizer
Checkpoint:
examples/circle_packing\openevolve_output\checkpoints\checkpoint_50
Branching
Performance
List
Metric: combined_score
Highlight: Top score
Dark mode:
[open in new window]
Program ID: 1bdab8d6-1f86-4973-b820-d886f0f99173
Island: 0
Generation: 3
Parent ID: 7069743d-68aa-4e07-9f1e-5df499030ca5 (island 0)
Metrics:
combined_score 0.8788
eval_time 0.4118
sum_radii 2.3156
target_ratio 0.8788
validity 1.0000
Code
Prompts
Diff
EVOLVE-BLOCK-START
"""Optimized circle packing for n=26
import numpy as np
def construct_packing():
"""
Construct a specific arrangement
that attempts to maximize the su
Strategy: Use a dense hexagonal
edge circles to utilize corner s