Documentation: Packing Fixed Polyominoes into the Smallest Possible Square

Artur Sahakyan

May 16, 2025

Introduction

This problem instantly caught my eye. It reminded me of the Tetris game. I was curious to seek a solution for it. For more visuals, please refer to the slides provided.

1 Tabu search method

This is a metaheuristics approach.

We start with lower bound for the side of 'possible' small square, S = $\sqrt{ShapeAreas}$ ($S \times S = ShapeAreas$). Tabu search will help us find a likely minimal packing, but not a provably minimal one. Nevertheless, we reason in terms of minimising a fitness function and starting the packing with the lower bound S. For better understanding let's go over the key steps:

- Random placement of shapes on valid positions
- Minimize 'illegal' cell count (cells that are out of bounds, overlap with other shapes). For that we define fitness = 1000 + illegal_cell_count: (some large number, say 1000 is used for imperfect solutions to be highlighted)
- Random shape is selected and put on a random legal position
- To avoid recent placements we define a TABU moves container (list in Python for instance). We keep the positions of recent moves and if they were illegal were put in TABU moves. But, if in upcoming solutions a move in TABU moves can improve the solution, we may get it out of the list and use it if better than the 'BEST so far' solution. [this helps us escape local optimas, search solution space and avoid infinite loops]
- If a neighbor solution is better (lower fitness) it is remembered as 'BEST so far' solution.
- Repeat the steps above until some stopping criterion, in this case number of iterations

2 Deterministic method (extra-result of research for solution)

This square packing problem can have a depth-first-search (DFS) and backtracking based method as well. Here, we again start with a lower bound S = $\sqrt{ShapeAreas}$. The logic is following

- Sort the shapes according to their area
- Place larger ones first. If the shapes don't fit increment the lower bound

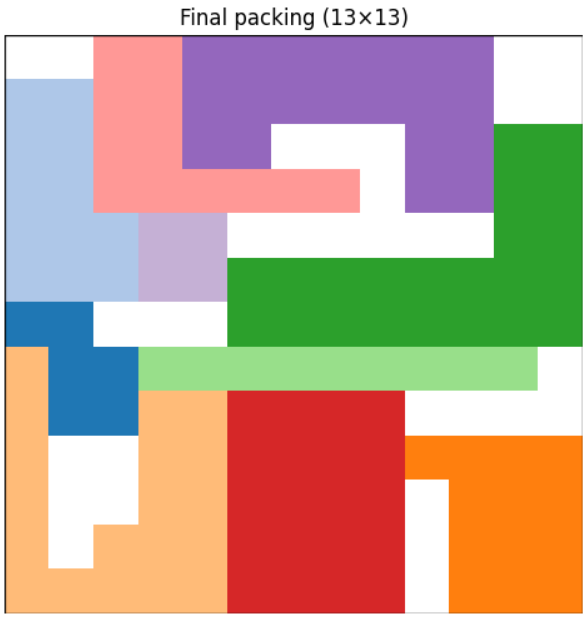

Figure 1: Result of packing with Tabu Search

- If all shapes fit $\implies$ success.
- If no success $\implies$ backtrack, remove the last shape and try placing in the next available position (we again take into account overlaps and the fact that rotations are not allowed)

3 Comparison

Deterministic solution completed in **7.021 seconds**.
Tabu Search completed in **4.631 seconds**. As we can observe, Tabu Search found a likely optimal packing faster. Sampling of solution space, avoiding cycles contributes to the speed of execution.