Dhanush Pyla

Insyde .IO – Assignment

Optimized Furniture Placement with a Genetic Algorithm

This document introduces a method for optimizing furniture placement within a small room through the use of a Genetic Algorithm (GA).

The aim is to have efficient furniture arrangement with minimal overlaps.
The procedure includes random initialization, evolution via selection and mutation, and fitness assessment to achieve the optimal layout.

Step-by-Step Explanation:

1. Define Room and Furniture Constraints

The room is delimited with a size of 10x6, and four pieces of furniture are taken into consideration, each having defined dimensions. These are a Bed, Sofa, Table, and Chair, each with the width and height defined.

2. Generate Initial Population (Random Layouts)

A first population of random furniture configurations is created.
Each configuration includes furniture at random locations within the room, making sure that all items stay within the boundaries set. This population is the foundation for the optimization process.

3. Define the Fitness Function (Overlap Calculation)

The main intention is to reduce overlapping furniture pieces. The fitness function compares the optimality of a particular furniture arrangement by counting the number of overlaps. It adds to the penalty for every overlapping pair, so the arrangement becomes less optimal.

The fitness function guides the genetic algorithm to better solutions by giving priority to arrangements with fewer or no overlaps.

4. Mutation Function (Small Positional Adjustments)

Mutation makes tiny random changes in the furniture locations, providing variability in the population. Mutation shifts furniture by a small amount within a given limit to try better locations.

This phase avoids local optima and improves the layouts across generations.

5. Genetic Algorithm (Evolution)

Optimization occurs according to the following steps:
Sorting layouts by fitness, favoring those with less overlap.
Maintaining the best 10 top layouts to ensure good solutions (elitism).
Creating new layouts through the mutation of best layouts until the population size is reestablished.
The cycle is repeated for a specified number of generations to locate the most optimal layout.

6. Visualizing the Optimized Layout

Once optimized, the optimal furniture arrangement is shown graphically. Every item of furniture is shown as a rectangle within the room, with its name.
This makes it simple to visualize the most effective arrangement.
Final Execution Process

The process consists of:

Creating an initial random population of furniture arrangements.
Developing the population through several generations.
Choosing the optimum layout according to the fitness function.
Showing the optimized layout graphically.

Conclusion

This technique optimizes furniture arrangement efficiently with Genetic Algorithms.
It makes sure that furniture is arranged within the room with reduced overlap.
The result offers a graphical output of the optimum furniture arrangement.

This strategy can be applied to bigger-scale optimization problems, including circuit board design, warehouse layout, and road network planning.

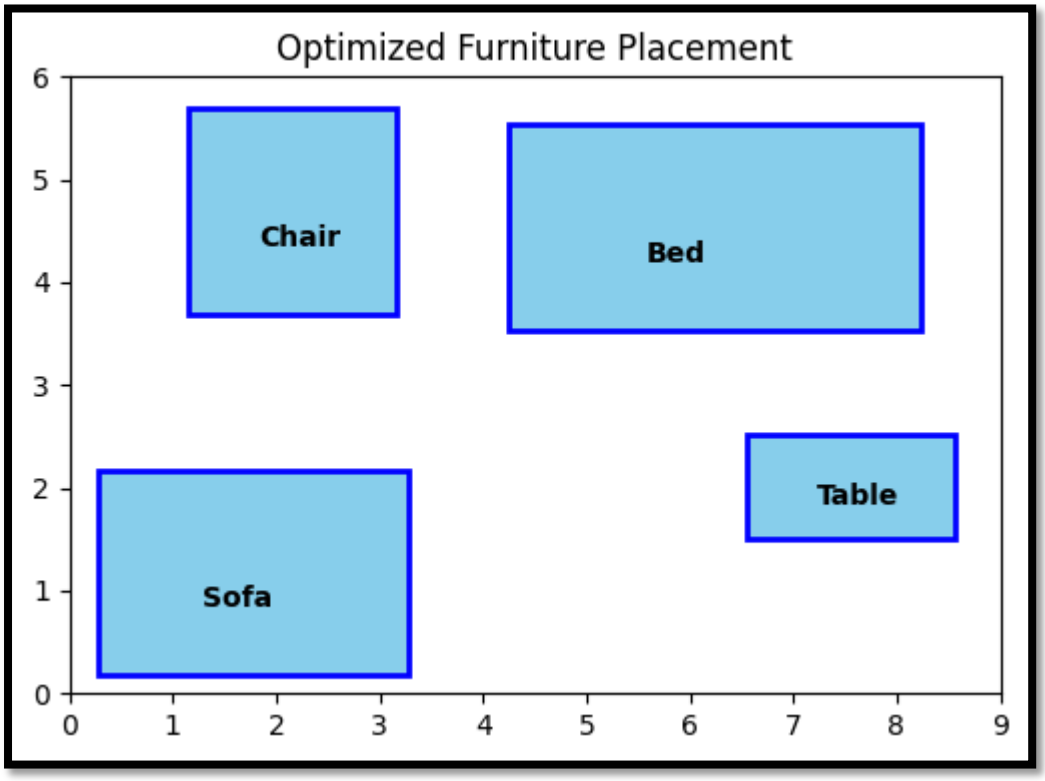
Code:

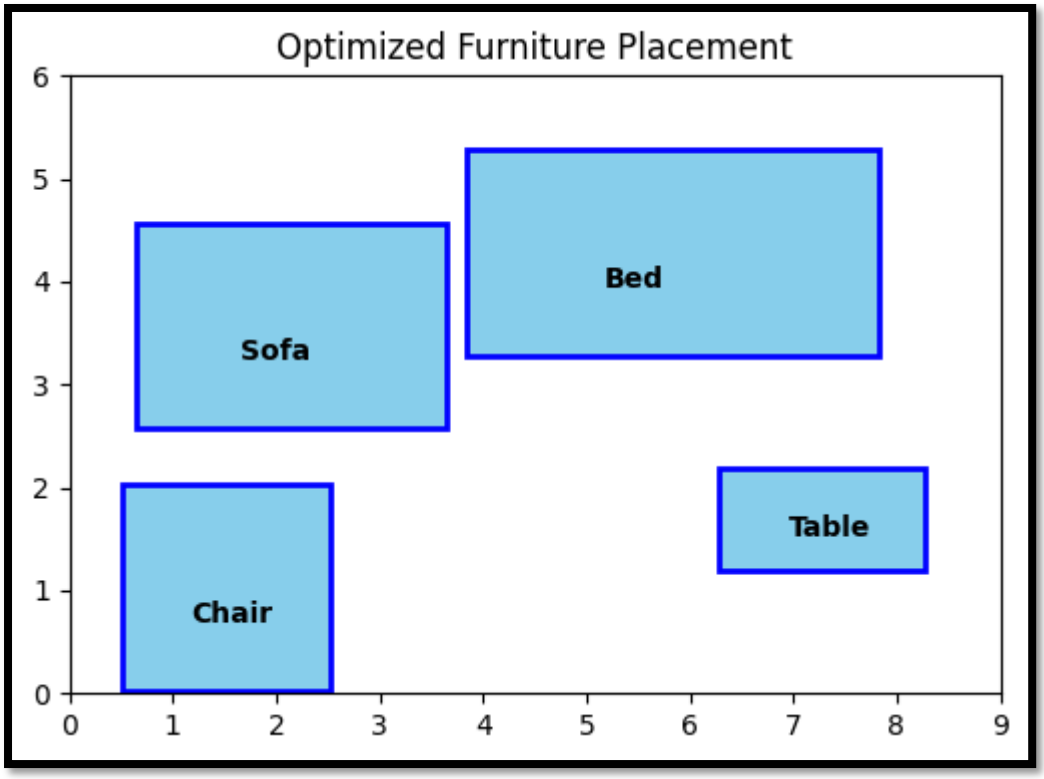
```
import numpy as np
import matplotlib.pyplot as plt

room_width = 9
room_height = 6

# Furniture items (name, width, height)
furniture = [
    ("Bed", 4, 2),
    ("Sofa", 3, 2),
    ("Table", 2, 1),
    ("Chair", 2, 2),
]

num_generations = 1000
population_size = 50
mutation_rate = 0.2

def generate_initial_population():
    """Generate random furniture placements inside room."""
    population = []
    for _ in range(population_size):
        layout = []
        for name, w, h in furniture:
            x = np.random.uniform(0, room_width - w)
            y = np.random.uniform(0, room_height - h)
            layout.append((name, w, h, x, y))
        population.append(layout)
```

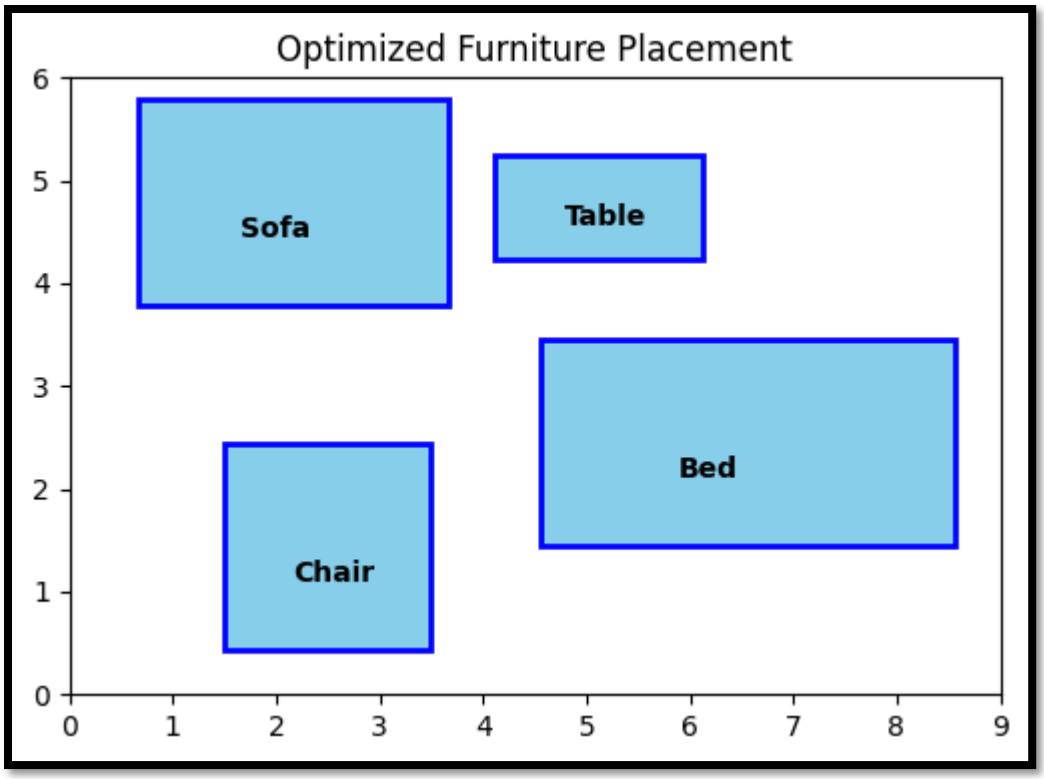
```
    return population

def check_overlap(f1, f2):
    """Returns True if two furniture pieces overlap."""
    _, w1, h1, x1, y1 = f1
    _, w2, h2, x2, y2 = f2

    return not (x1 + w1 <= x2 or x2 + w2 <= x1 or y1 + h1 <= y2 or y2 + h2 <= y1)

def fitness_function(layout):
    """Calculates fitness score. Lower is better."""
    overlap_penalty = 0
    for i in range(len(layout)):
        for j in range(i + 1, len(layout)):
            if check_overlap(layout[i], layout[j]):
                overlap_penalty += 1

    return overlap_penalty

def mutate(layout):
    """Mutate layout by slightly shifting furniture positions."""
    mutated_layout = []
    for item in layout:
        name, w, h, x, y = item
        if np.random.rand() < mutation_rate:
            x = max(0, min(room_width - w, x + np.random.uniform(-0.5, 0.5)))
            y = max(0, min(room_height - h, y + np.random.uniform(-0.5, 0.5)))
        mutated_layout.append((name, w, h, x, y))
    return mutated_layout
```

```
def evolve_population(population):
    """Run genetic algorithm for multiple generations."""
    for _ in range(num_generations):
        population.sort(key=fitness_function)
        new_population = population[:10]
        while len(new_population) < population_size:
            parent = population[np.random.randint(0, 10)]
            child = mutate(parent)
            new_population.append(child)
        population = new_population
    return population[0]

def plot_layout(layout):
    """Plot the optimized furniture placement."""
    fig, ax = plt.subplots(figsize=(6, 4))
    ax.set_xlim(0, room_width)
    ax.set_ylim(0, room_height)
    ax.set_title("Optimized Furniture Placement")

    for name, w, h, x, y in layout:
        ax.add_patch(plt.Rectangle((x, y), w, h, facecolor="skyblue", edgecolor="blue", lw=2))
        ax.text(x + w/3, y + h/3, name, fontsize=10, weight='bold')

    plt.show()

population = generate_initial_population()
best_layout = evolve_population(population)
plot_layout(best_layout)
```

Output:


Optimized Furniture Placement
Chair
Bed
Table
Sofa
0
1
2
3
4
5
6
7
8
9


Optimized Furniture Placement
Bed
Sofa
Table
Chair
0
1
2
3
4
5
6
7
8
9


Optimized Furniture Placement
Sofa
Table
Bed
Chair
0
1
2
3
4
5
6
7
8
9