Dokumentacja wstępna - POP

Maksymalne upakowanie kontenerów w magazynie o skończonej pojemności z wizualizacją rozkładu

Autorzy: Jakub Antas, Jerzy Muszyński

3 grudnia 2025

1 Opis problemu i koncepcja rozwiązania

Celem projektu jest rozwiązanie wariantu problemu ***3D Bin Packing***, w którym do dyspozycji jest jeden pojemnik (magazyn) o zadanych wymiarach, a zadanie polega na takim rozmieszczeniu zestawu kontenerów o różnych wymiarach, aby maksymalizować wykorzystanie objętości magazynu.

Użytkownik programu definiuje wejściowo listę kontenerów, z której każdy ma określoną długość, szerokość oraz wysokość. Zakładamy, że nie zawsze możliwe będzie umieszczenie wszystkich kontenerów w magazynie — zarówno ze względu na ograniczoną przestrzeń, jak i naturę zastosowanego algorytmu optymalizacyjnego.

1.1 Reprezentacja danych wejściowych

Dane wejściowe stanowią wektory opisujące każdy kontener w postaci trójki:

$$[l, w, h],$$

gdzie l oznacza długość, w szerokość, a h wysokość. Kolejność kontenerów w tym wektorze wejściowym jest stała i odpowiada kolejności kontenerów w rozwiązaniu.

1.2 Reprezentacja rozwiązania

Jedno rozwiązanie (osobnik w algorytmie ewolucyjnym) reprezentowane jest jako lista wektorów postaci:

$$[x, y, z, dx, dy, dz]$$

gdzie:

- (x, y, z) — współrzędne lewego, dolnego i najbliższego rogu danego kontenera w przestrzeni magazynu,
- (dx, dy, dz) — długości boków kontenera wzdłuż osi X, Y i Z, będące permutacją jego oryginalnych wymiarów (l, w, h).

Permutacja ta reprezentuje orientację kontenera. Przykładowo $(dx, dy, dz) = (w, l, h)$ odpowiada jednej z rotacji o 90 stopni.

Współrzędne przeciwnego rogu kontenera można otrzymać jako:

$$(x + dx, y + dy, z + dz).$$

Taki sposób zapisu jednoznacznie definiuje faktyczne rozmieszczenie wszystkich kontenerów w magazynie.

Kontenery niewykorzystane (nieupakowane)
Ze względu na to, że nie zawsze wszystkie kontenery zmieszczą się w magazynie, wprowadzono specjalne oznaczenie dla kontenerów, które algorytm postanowił pominąć. Taki „niewzięty" kontener reprezentowany jest wektorem:

$$[-1, -1, -1, -1, -1, -1]$$

Wartości -1 są sygnałem dla funkcji sprawdzającej, aby **ignorować** ten kontener. Dzięki temu nie powoduje on błędów kolizji ani wyjścia poza zakres, a po prostu nie wlicza się do wyniku punktowego.

1.3 Warunki poprawności rozwiązania

Rozwiązanie jest niedopuszczalne, jeśli występuje przynajmniej jedna z sytuacji:

1. **Kontener wystaje poza magazyn**, tzn. narusza warunek:

$$0 \leq x \leq x + dx \leq W_x,$$
$$0 \leq y \leq y + dy \leq W_y,$$
$$0 \leq z \leq z + dz \leq W_z.$$

2. **Dwa kontenery nachodzą na siebie** — zachodzi nakładanie przedziałów na wszystkich trzech osiach.

Warunek „braku lewitowania" (czyli potrzeby podparcia kontenera przez podłogę lub inne kontenery) w tej wersji projektu pomijamy, dopuszczając możliwość „podwieszenia" kontenerów. Można jednak łatwo dodać go w przyszłości.

1.4 Funkcja celu

Funkcją celu zastosowaną w projekcie jest: **łączna objętość kontenerów poprawnie umieszczonych w magazynie**.

Jeśli rozwiązanie jest niedopuszczalne, funkcja celu przyjmuje wartość 0, dzięki czemu wszystkie rozwiązania poprawne są automatycznie preferowane nad niepoprawnymi.

1.5 Algorytm optymalizacyjny

Rozwiązanie problemu upakowania 3D jest zadaniem bardzo złożonym obliczeniowo (tzw. problem NP-trudny). Dlatego w projekcie zastosujemy **algorytm ewolucyjny**, składający się z następujących etapów:

1. **Inicjalizacja** — wygenerowanie początkowej populacji losowych rozmieszczeń kontenerów.

2. **Selekcja** — wybór osobników o najwyższej wartości funkcji celu (np. selekcja progowa).

3. **Mutacja** — losowe modyfikacje, które wprowadzają różnorodność do populacji:

   - zmianę pozycji kontenerów: $x+ = \text{rand}(); \quad y+ = \text{rand}(); \quad z+ = \text{rand}();$
   - zmianę orientacji kontenerów poprzez losową permutację (dx, dy, dz) zgodną z oryginalnymi wymiarami.
   - upakowanie lub wypakowanie kontenera

4. **Krzyżowanie** — łączenie dwóch osobników (np. poprzez jednolite krzyżowanie), gdzie każdy kontener może zostać przejęty z jednego z rodziców.

5. **Sukcesja** — analiza starej populacji powiększonej o nową (po reprodukcji) w celu przekazania najlepszych osobników do kolejnych epok algorytmu

6. **Archiwizacja wyników** — zapisywanie wszystkich rozwiązań w historii, co pozwala ostatecznie wybrać najlepszy znaleziony osobnik.

2 Planowane eksperymenty numeryczne

W ramach analizy eksperymentalnej planowane jest zbadanie wpływu parametrów algorytmu ewolucyjnego na jakość otrzymywanych rozwiązań oraz szybkość zbieżności. Testowane będą między innymi:

- różne rozmiary populacji,
- różne prawdopodobieństwa mutacji,
- różne prawdopodobieństwa krzyżowania,
- wpływ typu selekcji (np. progowa vs. turniejowa),
- wpływ różnych intensywności mutacji (małe vs. duże zmiany pozycji).

3 Wybór technologii

Projekt planowany jest do realizacji w języku **Python**, który zapewnia odpowiednią elastyczność oraz dostęp do bibliotek wspierających obliczenia numeryczne i wizualizację.

W szczególności wykorzystywane będą:

- **NumPy** — do operacji matematycznych i manipulacji wektorami,
- **PyTorch** — jako wygodny framework do pracy na tensorach i potencjalnego przyspieszenia obliczeń (opcjonalnie),
- **Matplotlib** — do wizualizacji rozmieszczenia kontenerów w magazynie w postaci grafiki 3D,