Bin-Packing

Enache Stephan-Cristian, Ravariu Maria-Antonia, and Velicu Alexandru-Stefan[1]

Universitatea Nationala de Stiinta si Tehnologie Politehnica Bucuresti
https://upb.ro/

Abstract. In această lucrare am abordat problema NP-complete Bin-Packing. Aceasta presupune că avem n obiecte, fiecare de o anumită dimensiune, și câteva coșuri de gunoi de capacitate egală. Vrem să repartizăm obiectele în coșurile de gunoi, folosind cât mai puține coșuri posibil. Desigur, dimensiunea totală a obiectelor repartizate într-un coș de gunoi nu trebuie să depășească capacitatea acestuia. Am efectuat si o serie de teste care sa confirme corectitudinea si eficacitatea problemei.

Keywords: NP-complete · Bin-Packing · Grafuri.

1 Introducere

1.1 Descrierea problemei

Problema Bin-Packing este o problemă de optimizare în care articole de dimensiuni diferite trebuie ambalate într-un număr finit de containere, fiecare cu o capacitate fixă, astfel încât să se minimizeze numărul de containere utilizate. Această problemă are diverse aplicații, cum ar fi încărcarea containerelor, programarea sarcinilor și plasarea datelor pe mai multe discuri.

Definitia problemei Având n articole cu greutăți diferite și containere cu capacitate c, obiectivul este de a aloca fiecare articol unui container astfel încât numărul total de containere utilizate să fie minimizat. Se presupune că toate articolele au greutăți mai mici decât capacitatea containerului.

Se da un sir de numere pozitive $a_1, \ldots, a_n \leq 1$.

Se identifica $k \in \mathbb{N}$ și o funcție de asignare

$$f : \{1, \ldots, n\} \to \{1, \ldots, k\}$$

astfel încât, pentru fiecare $j \in \{1, \ldots, k\}$, să se verifice

$$\sum_{\substack{i=1 \\ f(i)=j}}^{n} a_i \leq 1,$$

iar k să fie minim posibil.

1.2 Heuristici pentru Bin Packing

Ipoteza Problema de bin packing este NP-hard, deci este improbabil să existe un algoritm polinomial care să producă întotdeauna soluția optimă. Din acest motiv, folosim *heuristici* care oferă soluții bune, de obicei apropiate de optim.

Algoritmul Next-Fit (NF) Idee:

- Fiecare obiect este încercat în *bin-ul curent.*
- Dacă încape, este plasat acolo.
- Dacă nu încape, se deschide un bin nou.

Next Fit Packing

Raport de aproximație pentru Next-Fit Fie M numărul minim de bin-uri necesare pentru o instanță I.

Theorem 1. *Algoritmul Next Fit folosește cel mult* $2M$ *bin-uri.*

Proof. Fie $s(B_i)$ suma dimensiunilor obiectelor din bin-ul B_i în soluția NF. Pentru orice doi biți consecutivi B_j și B_{j+1} avem:

$$s(B_j) + s(B_{j+1}) > 1.$$

Dacă NF folosește k bin-uri (presupunem k par), atunci:

$$\sum_{i=1}^{k} s(B_i) > \frac{k}{2}.$$

Dar suma totală a obiectelor este $\leq M$, deci:

$$\frac{k}{2} < M \quad \Rightarrow \quad k < 2M.$$

Limită inferioară pentru Next-Fit Există secvențe de intrare pentru care Next Fit folosește $2M - 2$ bin-uri, unde M este optimul.

1.3 Algoritmul First-Fit (FF)

- Se scanează bin-urile în ordine.
- Obiectul este plasat în *primul bin* în care încape.
- Se creează un bin nou doar dacă nu încape în niciunul dintre cele existente.

First Fit Packing

Complexitate

- Implementare simplă: $O(n^2)$.
- Implementare rapidă: $O(n \log n)$ folosind un arbore de căutare echilibrat.

Structura nodului arborelui:

- indexul bin-ului,
- capacitatea rămasă,
- valoarea maximă din subarbore.

Raport de aproximație pentru First-Fit

Theorem 2. *Dacă M este numărul optim de bin-uri, First Fit folosește cel mult $\lceil 1.7M \rceil$ bin-uri.*

Există secvențe pentru care First Fit folosește $1.6666 \ldots M$ bin-uri.

Limită inferioară Există cazuri în care First Fit folosește $10M$ bin-uri, în timp ce soluția optimă folosește doar $6M$.

1.4 Algoritmul Best-Fit (BF)

Idee:

- Obiectul este plasat în bin-ul în care încape *cel mai strâns* (adică rămâne cel mai puțin spațiu liber).
- Dacă nu încape în niciun bin, se creează unul nou.

Complexitate: $O(n \log n)$ folosind un arbore echilibrat ordonat după capacitatea rămasă.

2 Heuristici derivate

- **First Fit Decreasing (FFD)**: se sortează obiectele descrescător, apoi se aplică FF.
- **Best Fit Decreasing (BFD)**: se sortează obiectele descrescător, apoi se aplică BF.

Exemple aplecatii practice Nu există multe probleme de optimizare combinatorie a căror relevanță practică să fie mai evidentă. De exemplu, cea mai simplă versiune a problemei CuttingStock este echivalentă: ni se dau multe grinzi de lungime egală (să zicem 1 metru) și numere $a_1, ..., a_n$. Vrem să tăiem cât mai puține grinzi în bucăți, astfel încât la final să avem grinzi de lungimi $a_1, ..., a_n$.

2.1 Aplicarea problemei *Bin Packing* la realizarea unui mix-tape

Problema realizării unui *mix-tape* poate fi modelată în mod natural ca o instanță a problemei clasice de *bin packing*. În acest context:

- fiecare melodie reprezintă un *obiect* (item);
- durata melodiei reprezintă *greutatea* obiectului;
- fiecare casetă (sau fiecare parte a casetei) reprezintă un *bin*;
- capacitatea unui bin este durata maximă care poate fi înregistrată (de exemplu, 30 sau 60 de minute).

Modelare formală Fie o listă de melodii cu durate

$$a_1, a_2, \ldots, a_n, \qquad a_i > 0.$$

Fiecare melodie trebuie plasată într-un bin (casetă) astfel încât suma duratelor din fiecare bin să nu depășească capacitatea C:

$$\sum_{i:f(i)=j} a_i \leq C, \qquad \forall j.$$

Obiectivul este minimizarea numărului total de bin-uri:

$$\min k.$$

Aceasta este exact formularea problemei de *bin packing*.

Interpretare intuitivă Realizarea unui mix-tape presupune alegerea unui set de melodii care să încapă într-o durată fixă. Dacă numărul melodiilor este mare sau dacă se dorește realizarea mai multor mixuri, apare necesitatea împărțirii lor în mai multe casete. Astfel, problema devine:

> *Cum putem împacheta melodiile în cât mai puține casete, fără a depăși durata maximă a fiecăreia?*

Heuristici utilizabile Heuristicile clasice de bin packing se aplică direct:

- **Next Fit**: se adaugă melodiile în caseta curentă până nu mai încape nimic, apoi se trece la următoarea.
- **First Fit**: fiecare melodie este plasată în prima casetă în care încape.
- **Best Fit**: melodia este plasată în caseta în care rămâne cel mai puțin spațiu liber după adăugare.
- **First Fit Decreasing / Best Fit Decreasing**: melodiile sunt sortate descrescător după durată înainte de împachetare, ceea ce produce rezultate mult mai bune în practică.

3 Reducerea Partition la Bin Packing (decizie)

3.1 Problema Partition

Fie o mulțime A de numere pozitive. Putem împărți A în două submulțimi disjuncte A_1 și A_2 astfel încât:

$$\sum_{a_i \in A_1} a_i = \sum_{a_j \in A_2} a_j?$$

4 Demonstrație NP-Hard

4.1 Problema Bin Packing (decizie)

Fie o mulțime B de N elemente pozitive și o capacitate C. Putem distribui elementele în K binuri astfel încât:

$$\sum_{b_i \in B_j} b_i \leq C, \quad \text{pentru fiecare bin } B_j?$$

4.2 Reducere

Construim o instanță de Bin Packing pornind de la Partition:

- $B = A$ (elementele rămân neschimbate)
- $C = \frac{\sum a_i}{2}$
- $K = 2$

Dacă Partition are soluție, atunci există două submulțimi A_1 și A_2 cu sume egale:

$$\sum_{a_i \in A_1} = \sum_{a_j \in A_2} = \frac{\sum a_k}{2}$$

$\Rightarrow$ putem distribui elementele în două binuri de capacitate $C \Rightarrow$ Bin Packing = TRUE.

Analog, dacă Bin Packing are soluție cu $K = 2$ și $C = \frac{\sum a_i}{2}$, atunci cele două binuri sunt umplute exact $\Rightarrow$ sumele sunt egale $\Rightarrow$ Partition = TRUE.

4.3 Concluzie

$$\text{Partition} \leq_p \text{Bin Packing (decizie)}$$

—

5 Reducerea 3-Partition la Bin Packing (decizie)

5.1 Problema 3-Partition

Fie o mulțime A de numere pozitive, cu $|A| = 3N$. Putem împărți A în N triplete disjuncte astfel încât fiecare triplet să aibă aceeași sumă?

$$S_{\text{triplet}} = \frac{\sum a_i}{N}$$

5.2 Construirea instanței de Bin Packing

- $B = A$
- $C = \frac{\sum a_i}{N} = S_{\text{triplet}}$
- $K = N = \frac{|B|}{3}$

5.3 Implicație

"$\Leftarrow$":

Dacă 3-Partition are soluție $\Rightarrow$ putem grupa elementele în N triplete de sumă $C \Rightarrow$ fiecare triplet intră într-un bin, ocupându-l în totalitate $\Rightarrow$ Bin Packing are soluție cu $K = N$.

"$\Rightarrow$":

Analog, dacă Bin Packing are soluție cu $K = N$ și capacitate C, iar suma totală este $N \cdot C$, atunci fiecare bin este umplut cu exact C, iar fiecare bin conține exact 3 elemente $\Rightarrow$ avem o soluție pentru 3-Partition.

5.4 Concluzie

$$\text{3-Partition} \leq_p \text{Bin Packing (decizie)}$$

Bin Packing (decizie) este NP-completă.