

Bin Packing Problem - NP-Hard

Nicolas Lian Yang Chavez
Universidad Nacional de Itapúa
Encarnación, Paraguay
lian.yang@fiumi.edu.py

Genaro Nieto Matias Brunaga
Karajallo
Universidad Nacional de Itapúa
Encarnación, Paraguay
genaro.brunaga@fiumi.edu.py

Sebastian Simeon Bogado Mohr
Universidad Nacional de Itapúa
Encarnación, Paraguay
sebastian.bogado2019@fiumi.edu.py

Resumen

El problema de empaquetado (Bin Packing Problem) es un problema clásico de optimización combinatoria NP-Hard que consiste en asignar un conjunto de objetos de diferentes tamaños a un número mínimo de contenedores de capacidad fija. Este artículo presenta una revisión de los principales enfoques para resolver el BPP en 1D, clasificados en tres categorías: métodos exactos, heurísticos y meta-heurísticos. Analizamos Branch and Bound como método exacto garantizando soluciones óptimas con limitación de escalabilidad. Examinamos métodos heurísticos simples (Next Fit, First Fit, Best Fit) y con ordenamiento (FFD, BFD), proporcionando análisis de complejidad y garantías de aproximación. Finalmente, exploramos algoritmos genéticos como meta-heurística para instancias de gran escala.

CCS Concepts

• **Computing methodologies** → Genetic algorithms; • **Theory of computation** → Packing and covering problems; *Approximation algorithms analysis*.

Keywords

Problema de empaquetado, optimización combinatoria, NP-Hard, ramificación y poda, algoritmos heurísticos, First Fit, Best Fit, algoritmos genéticos, algoritmos de aproximación, complejidad computacional

ACM Reference Format:

Nicolas Lian Yang Chavez, Genaro Nieto Matias Brunaga Karajallo, and Sebastian Simeon Bogado Mohr. 2025. Bin Packing Problem - NP-Hard. In *Proceedings of Bin Packing Problem: Análisis Comparativo de Métodos de Solución (BPP Research)*. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/nnnnnnnn.nnnnnnnn>

1. Introducción

El Bin Packing Problem (BPP) es un problema NP-Hard fundamental en optimización combinatoria con amplia aplicabilidad en industria y computación. Este artículo examina los principales enfoques desarrollados para su resolución, desde métodos exactos hasta meta-heurísticas modernas, proporcionando un análisis comparativo que facilita la selección del método apropiado según las características del problema y restricciones computacionales.

1.1. Formulación del Problema

Sea $I = \{i_1, i_2, \dots, i_n\}$ un conjunto de n objetos, donde cada objeto i_j tiene un tamaño $s_j \in (0, C]$. Sea B un conjunto ilimitado

de contenedores idénticos con capacidad C . El BPP consiste en asignar todos los objetos a un número mínimo de contenedores tal que la suma de tamaños no exceda la capacidad.

Formulación Matemática:

$$\text{Minimizar: } m \text{ (número de contenedores)} \quad (1)$$

Sujeto a:

$$\sum_{j=1}^n s_j \cdot x_{ij} \leq C, \quad \forall i = 1, \dots, m \quad (2)$$

$$\sum_{i=1}^m x_{ij} = 1, \quad \forall j = 1, \dots, n \quad (3)$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \quad (4)$$

Donde $x_{ij} = 1$ si el objeto j está en el contenedor i . La cota inferior teórica es:

$$L_0 = \left\lceil \frac{\sum_{j=1}^n s_j}{C} \right\rceil \quad (5)$$

2. Estado del Arte

El Bin Packing Problem fue formalizado por Johnson et al. en 1974, demostrando su pertenencia a la clase NP-Hard. Esta clasificación tuvo profundas implicaciones: no existe (bajo la conjetura $P \neq NP$) un algoritmo polinomial que resuelva todas las instancias óptimamente. Esta realidad impulsó tres décadas de investigación en enfoques alternativos.

Década de 1970: Fundamentos teóricos y primeros algoritmos heurísticos (Ullman 1971, Johnson 1973). Análisis de garantías de aproximación para Next Fit, First Fit y Best Fit.

Décadas de 1980-1990: Refinamiento de heurísticas con ordenamiento previo. Martello y Toth (1990) desarrollan métodos exactos optimizados. Se establecen cotas de aproximación mejoradas: FFD alcanza $\frac{11}{9}OPT + \frac{6}{9}$.

2000-presente: Aplicación de meta-heurísticas (algoritmos genéticos, simulated annealing, tabu search) y métodos híbridos. Integración con aprendizaje automático para casos específicos.

3. Enfoques de Solucion

Los métodos de solución se clasifican en: (1) **Exactos** (Branch and Bound, programación lineal entera): garantizan optimalidad pero son computacionalmente costosos; (2) **Heurísticos** (Next Fit, First Fit, Best Fit, FFD, BFD): ofrecen soluciones rápidas con garantías de aproximación; (3) **Meta-heurísticos** (algoritmos genéticos, simulated annealing): exploran el espacio de soluciones de forma inteligente para instancias grandes sin garantía de optimalidad.

3.1. Métodos Exactos: Branch and Bound

Branch and Bound (B&B) es un método de búsqueda exhaustiva que explora sistemáticamente el espacio de soluciones mediante un árbol de decisiones, garantizando la solución óptima. El algoritmo construye el árbol ramificando en cada nodo las posibles asignaciones de objetos a contenedores, y poda ramas que no pueden mejorar la mejor solución conocida.

Funcionamiento: Cada nodo representa un estado parcial de asignación. Para cada objeto sin asignar, se generan ramas correspondientes a colocarlo en contenedores existentes (si cabe) o en uno nuevo. Se mantiene el mejor número de contenedores encontrado como cota superior. Una rama se poda si el número de contenedores ya utilizados más los mínimos necesarios para los objetos restantes excede la cota superior.

Garantía de Optimalidad: B&B garantiza optimalidad por: (1) exploración completa del espacio factible, (2) poda conservadora que solo elimina ramas demostrablemente subóptimas, y (3) actualización monotónica de la mejor solución. Si existe una solución con k bins, el algoritmo la encontrará.

Limitación de Escalabilidad: El árbol de decisiones crece exponencialmente ($O(n^n)$ nodos potenciales). Para $n = 20$ objetos, el espacio de búsqueda alcanza 10^{20} nodos, haciendo el tiempo de ejecución inviable. Aunque la poda reduce significativamente los nodos explorados, la reducción es insuficiente para compensar el crecimiento exponencial. B&B es práctico solo para instancias pequeñas ($n \leq 25$) donde se requiere optimalidad certificada. Para aplicaciones industriales con cientos o miles de objetos, los métodos heurísticos y meta-heurísticos son indispensables.

3.2. Métodos Heurísticos

3.2.1. Métodos Simples. Los métodos heurísticos simples para el *Bin Packing Problem* son algoritmos de tipo *greedy* que procesan los ítems secuencialmente, colocándolos en un contenedor según una regla local sin reordenamientos ni retrocesos. No garantizan la optimalidad, pero ofrecen soluciones eficientes en tiempo y espacio.

- Se mantienen contenedores abiertos con capacidad restante.
- Cada ítem se coloca según el criterio del método.
- Si no cabe en ninguno, se abre un nuevo contenedor.

Next Fit (NF). Mantiene un único contenedor abierto. Si el ítem cabe, se coloca; de lo contrario, se cierra el contenedor y se abre uno nuevo. **Complejidad:** $O(n)$ tiempo, $O(1)$ memoria.

$$NF(L) \leq 2 \cdot OPT(L) - 1,$$

First Fit (FF). Mantiene una lista de contenedores abiertos. Cada ítem se coloca en el primer contenedor donde quepa; si no, se abre uno nuevo. **Complejidad:** $O(n^2)$

Best Fit (BF). Coloca cada ítem en el contenedor que deje el menor espacio libre posible. Si no cabe en ninguno, se abre uno nuevo. **Complejidad:** $O(n^2)$

Cotas de aproximación (First Fit y Next Fit):

$$FF(L), BF(L) \leq \lfloor 1.7 \cdot OPT(L) \rfloor$$

3.2.2. Métodos con Ordenamiento Previo. Estos métodos aplican la misma lógica que los anteriores, pero primero ordenan los ítems en forma **decreciente** según su tamaño antes de ejecutar la heurística.

Este ordenamiento reduce la fragmentación y mejora la utilización del espacio.

First Fit Decreasing (FFD). Ordena los ítems de mayor a menor y aplica el método *First Fit*. **Complejidad:** $O(n \log n)$.

Best Fit Decreasing (BFD). Ordena los ítems de forma descendente y aplica *Best Fit*, minimizando el espacio libre restante. **Complejidad:** $O(n \log n)$.

Cota de aproximación:

$$FFD(L), BFD(L) \leq \frac{11}{9} \cdot OPT(L) + 1$$

3.3. Métodos Meta-Heurísticos

Los métodos metaheurísticos constituyen un conjunto de estrategias de optimización aproximada que buscan soluciones cercanas al óptimo en problemas de alta complejidad computacional, especialmente en aquellos pertenecientes a la clase NP-difícil, donde los métodos exactos resultan inviables debido al crecimiento exponencial del espacio de búsqueda. A diferencia de los métodos heurísticos clásicos —que suelen diseñarse para un problema específico y proporcionan soluciones rápidas mediante reglas empíricas—, los metaheurísticos son marcos generales de búsqueda estocástica capaces de adaptarse a una amplia variedad de problemas de optimización combinatoria.

El algoritmo genético modela el proceso de selección natural. Cada solución candidata (denominada cromosoma) representa una forma de empaquetar los objetos en los contenedores. A partir de una población inicial generada aleatoriamente, se aplican operadores de **selección**, **cruce** y **mutación** para explorar el espacio de soluciones. El operador de selección elige los individuos más aptos según una función objetivo, comúnmente el número total de contenedores utilizados. Los operadores de cruce y mutación introducen variabilidad, combinando características de soluciones previas o alterando partes de ellas, lo que evita la convergencia prematura hacia óptimos locales.

A continuación, se provee el código implementado en Python para la optimización de [...]:

4. Conclusión

El Bin Packing Problem representa un desafío fundamental en optimización combinatoria cuya naturaleza NP-Hard ha motivado el desarrollo de múltiples enfoques de solución adaptados a diferentes contextos.

Los métodos exactos como Branch and Bound son óptimos cuando se requiere garantía de solución óptima en instancias pequeñas ($n \leq 25$ objetos), donde el costo exponencial es manejable. Para instancias medianas (30-1000 objetos), los heurísticos FFD y BFD representan el balance ideal: soluciones de alta calidad con garantías de aproximación ($\frac{11}{9}OPT$) en tiempo cuasi-lineal ($O(n \log n)$), siendo la opción práctica para aplicaciones industriales. Los algoritmos genéticos son apropiados para grandes volúmenes ($n > 1000$), donde su exploración estocástica permite encontrar configuraciones de alta calidad inaccesibles mediante métodos determinísticos.

La selección del método debe fundamentarse en el tamaño de la instancia y los requisitos de optimalidad versus tiempo de ejecución del contexto aplicado.