Polynomial-Time Approximation Schemes for the 1-D Bin Packing Problem

Pranav Swarup Kumar

November 9, 2025

Contents

1 Introduction (1.1)

The bin packing problem tries to minimize the number of unit-capacity bins required to pack n items of sizes $s_1, s_2, \ldots, s_n \in (0, 1]$.
In this section, 1-D bin packing is proved to be NP-Hard by reduction from the Partition Problem. *Approximation algorithms* which are extensively used to produce near-optimal solutions, are defined and discussed.

2 Proof of NP-hardness (1.2)

Optimization Problem Definition

The **one-dimensional Bin Packing Problem (BPP)** can be stated as follows.

Instance. A finite multiset of items

$$S = \{s_1, s_2, \ldots, s_n\}, \quad s_i \in (0, 1].$$

Each s_i denotes the size of item i, and each bin has unit capacity.

Objective. Partition S into the minimum number of disjoint subsets (bins)

$$B_1, B_2, \ldots, B_m$$

such that, for every bin B_j,

$$\sum_{s_i \in B_j} s_i \leq 1.$$

The goal is to minimize m. We denote the optimal number of bins by

$$OPT(S) = \min\{\, m \mid \exists\, B_1, \ldots, B_m \text{ satisfying the above constraint} \,\}.$$

Decision Problem Definition

To analyze computational complexity, the optimization problem is converted into a decision problem, which asks a yes/no question instead of minimizing a quantity.

Formally, the decision version of the 1-D Bin Packing Problem is defined as follows.

Instance. A multiset of items

$$S = \{s_1, s_2, \ldots, s_n\}, \quad s_i \in (0, 1],$$

and an integer $k > 0$.

Question. Does there exist a feasible packing of the items into at most k bins such that

$$\forall j \in \{1, \ldots, k\}, \quad \sum_{s_i \in B_j} s_i \le 1,$$

and every item appears in exactly one bin?

We denote this decision problem as BIN-PACK-DEC. Formally:

$$\text{BIN-PACK-DEC}(S, k) = \begin{cases} 1, & \text{if } \exists\{B_1, \ldots, B_k\} \text{ with } \sum_{s_i \in B_j} s_i \le 1, \\ 0, & \text{otherwise.} \end{cases}$$

Relationship between Optimization and Decision Versions. The optimization problem and its decision version are *polynomially equivalent* in the sense that

$$OPT(S) = \min\{\, k \mid \text{BIN-PACK-DEC}(S, k) = \text{“YES”} \,\}.$$

Hence, if the decision problem could be solved in polynomial time, the optimal packing number could be obtained by binary search over $k \in \{1, \ldots, n\}$, implying a polynomial-time algorithm for the optimization form. Therefore, the decision version is NP-complete if and only if the optimization version is NP-hard.

Reduction from Subset-Sum to Partition to Bin-Pack-Dec

We establish the computational intractability of the one-dimensional Bin Packing problem by a chain of polynomial-time reductions. We first reduce the classical SUBSET-SUM decision problem to PARTITION, and then reduce PARTITION to the decision version of Bin Packing (BIN-PACK-DEC). By transitivity of polynomial reductions and standard NP-completeness results, this shows that BIN-PACK-DEC is NP-complete and the optimization version of Bin Packing is NP-hard.

Reduction Subset-Sum $\leq_p$ Partition.

Let $(a_1, \ldots, a_n; t)$ be an instance of SUBSET-SUM, and denote $A := \sum_{i=1}^{n} a_i$.

Preprocessing. If $t > A$ then the instance is trivially a NO instance; return any fixed NO instance of PARTITION. Otherwise, if $t > A/2$ replace t by $A - t$. This substitution is valid since a subset sums to t iff its complement sums to $A - t$. After this step we have $0 \leq t \leq A/2$.

Construction. Define an instance of PARTITION by appending a single integer

$$b \;:=\; A - 2t \;\in\; \mathbb{Z}_{\geq 0}$$

to the original multiset. That is, form

$$S' \;=\; \{a_1, \ldots, a_n,\, b\}.$$

The total sum of S' is

$$A' \;=\; \sum_{i=1}^{n} a_i + b \;=\; A + (A - 2t) \;=\; 2(A - t),$$

hence A' is even and $A'/2 = A - t$.

Correctness.

(⇒) If there exists $I \subseteq \{1, \ldots, n\}$ with $\sum_{i \in I} a_i = t$, then $I' := I \cup \{b\}$ satisfies

$$\sum_{i \in I'} s_i = t + (A - 2t) = A - t = A'/2,$$

so S' admits a partition into two equal-sum subsets.

(⇐) Conversely, suppose S' admits a partition into two subsets of sum $A'/2 = A - t$. The element b must lie in one of the two parts; removing b from that part yields a subset of $\{a_1, \ldots, a_n\}$ whose sum is

$$(A - t) - b = (A - t) - (A - 2t) = t,$$

hence the original SUBSET-SUM instance is a YES instance.

The mapping adds one integer and performs only arithmetic operations on the input integers; thus it runs in polynomial time and preserves YES/NO answers. Therefore SUBSET-SUM $\leq_p$ PARTITION.

Reduction Partition $\leq_p$ Bin-Pack-Dec.

Let $(a_1, \dots, a_n)$ be an instance of PARTITION and set $A := \sum_{i=1}^{n} a_i$.

Preprocessing. If there exists i with $a_i > A/2$ then the PARTITION instance is immediately NO; return a fixed NO instance of BIN-PACK-DEC. Otherwise all $a_i \leq A/2$.

Construction. Create a Bin Packing instance by scaling:

$$s_i := \frac{2a_i}{A} \in (0, 1], \qquad i = 1, \dots, n,$$

and set $k := 2$. Note that

$$\sum_{i=1}^{n} s_i = \frac{2}{A} \sum_{i=1}^{n} a_i = 2.$$

Correctness.

(⇒) If the a_i admit a partition I with $\sum_{i \in I} a_i = A/2$, then the corresponding items satisfy

$$\sum_{i \in I} s_i = \frac{2}{A} \sum_{i \in I} a_i = 1,$$

so I and its complement form two bins of capacity 1 and the Bin Packing decision instance is YES.

(⇐) Conversely, if the s_i can be packed into two unit bins, each bin must have total size exactly 1 (since the grand total is 2). Scaling back by $A/2$ yields a partition of the a_i into two subsets of sum $A/2$.

The scaling mapping is computable in polynomial time (rational arithmetic) and preserves YES/NO answers, hence PARTITION $\leq_p$ BIN-PACK-DEC.

Transitivity and conclusion.

Polynomial-time reducibility is transitive. Combining the two reductions above we obtain

$$\textsc{Subset-Sum} \leq_p \textsc{Partition} \leq_p \textsc{Bin-Pack-Dec},$$

whence SUBSET-SUM $\leq_p$ BIN-PACK-DEC. Since SUBSET-SUM is NP-complete (see [3]), BIN-PACK-DEC is NP-hard. As BIN-PACK-DEC is in NP (a packing into k bins is polynomially verifiable), it follows that BIN-PACK-DEC is NP-complete (see [1]). Consequently the optimization form of one-dimensional Bin Packing is NP-hard.

Remarks on edge cases and polynomiality

The reductions above include simple preprocessing to handle degenerate inputs. In the reduction Subset-Sum $\leq_p$ Partition we replace the target t by $A - t$ when $t > A/2$; this step is valid because a subset sums to t iff its complement sums to $A - t$. If $t > A$ the SUBSET-SUM instance is trivially NO. The appended integer $b = A - 2t$ is non-negative by construction; if $b = 0$ the appended zero does not alter partitionability. In the reduction Partition $\leq_p$ Bin-Pack-Dec we first check whether any $a_i > A/2$: if so, PARTITION is immediately NO and we may output a fixed NO instance of BIN-PACK-DEC. Otherwise every scaled size $s_i = 2a_i/A$ satisfies $0 < s_i \leq 1$, so the constructed Bin Packing instance is valid. All arithmetic performed (sums, subtraction, a single division by A) is polynomial-time with respect to the binary encoding of the input integers; the bit-length of intermediate integers remains polynomially bounded. Thus both mappings are polynomial-time reductions in the standard Turing model.

3 Approximation Algorithms

An algorithm A is a ρ-approximation if for all instances I,

$$\frac{A(I)}{OPT(I)} \leq \rho$$

for minimization problems.

Examples

- First-Fit (FF): uses at most $1.7 \times OPT$ bins.
- Best-Fit (BF): similar constant-factor guarantee.

4 PTAS and FPTAS (1.4)

PTAS Definition

A **Polynomial-Time Approximation Scheme (PTAS)** is a family of algorithms $\{A_\varepsilon\}$ such that for any fixed $\varepsilon > 0$,

$$A_\varepsilon(I) \leq (1 + \varepsilon)\, OPT(I)$$

and A_ε runs in time polynomial in n (but possibly exponential in $1/\varepsilon$).

FPTAS Definition

An **FPTAS** is a PTAS whose running time is polynomial in both n and $1/\varepsilon$.

Impossibility Result

There is no FPTAS for Bin Packing unless P = NP [1].

5 APTAS (1.5)

Formal Definition

An **Asymptotic PTAS (APTAS)** satisfies:

$$A_\varepsilon(I) \le (1+\varepsilon)\, OPT(I) + O(1)$$

Intuitively, the additive $O(1)$ term is negligible for large instances, making APTAS the right notion for Bin Packing.

6 Classic APTAS: Fernández-de la Vega & Lueker (1981) (1.6)

Theorem 1 (Fernández-de la Vega and Lueker, 1981)**.** *For any $\varepsilon > 0$, there exists a polynomial-time algorithm A_ε such that*

$$A_\varepsilon(I) \le (1+\varepsilon)\, OPT(I) + 1.$$

Algorithm Sketch.

1. Classify items into large and small.
2. Round large items to few distinct sizes.
3. Enumerate all feasible bin patterns.
4. Fill remaining space with small items using First-Fit.

□

7 Variants and Improvements (1.7)

Briefly describe robust APTAS (Epstein & Levin), AFPTAS, variable-sized bins, and LP-based additive-gap algorithms by Hoberg & Rothvoss.

8 Worked Example (1.8)

Table 1: Example Bin Packing Instance

Item	Size	FF Bin	APTAS Bin
1	0.55	1	1
2	0.45	2	1
3	0.60	3	2
4	0.40	3	2

Table 2: Comparison of Algorithms

Algorithm	Approx. Ratio	Additive Gap	Typical Use
FF / BF / FFD / BFD	Constant	$O(OPT)$	Simple heuristics
APTAS	$(1+\varepsilon)$	$+O(1)$	Theoretical + practical
Rothvoss (LP)	$(1+\varepsilon)$	$+O(\log OPT)$	Very large n
KK / CKK	Heuristic	—	Partition / subset-sum

9 Comparative Discussion (1.9)

10 Advanced Algorithms (1.10)

Summarize Karmarkar–Karp (1982), Rothvoss (2014), and Hoberg–Rothvoss (2021) additive-gap LP algorithms.

References

[1] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, 1979.

[2] W. Fernandez de la Vega and G. Lueker, "Bin packing can be solved within $1 + \epsilon$ in linear time," *Combinatorica*, vol. 1, no. 4, pp. 349–355, 1981.

[3] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher (eds.), Plenum Press, New York, 1972, pp. 85–103.

[4] N. Karmarkar and R. M. Karp, "The differencing method of set partitioning," *Computer Science Division*, Univ. of California, Berkeley, Tech. Rep. UCB/CSD 82-113, 1982.

[5] E. G. Coffman Jr., M. R. Garey, and D. S. Johnson, "Approximation algorithms for bin packing: a survey," in *Approximation Algorithms for NP-Hard Problems*, D. S. Hochbaum (ed.), PWS Publishing, 1996, pp. 46–93.

[6] L. Epstein and A. Levin, "On bin packing with a constant number of bins," *SIAM Journal on Computing*, vol. 41, no. 6, pp. 1675–1697, 2012.