# The Martello-Toth Procedure (MTP) for the One-Dimensional Bin Packing Problem

A Detailed Algorithmic Analysis

December 1, 2025

**Abstract**

The One-Dimensional Bin Packing Problem (1DBPP) is an NP-hard combinatorial optimization problem. While heuristic approaches provide approximate solutions, exact algorithms are required to determine the minimal number of bins $m$. This document details the Martello-Toth Procedure (MTP), an exact Branch-and-Bound algorithm. We rigorously define the bounding procedures ($L_1$ and $L_2$), the dominance criteria used for problem reduction, and provide the mathematical proof of correctness for the $L_2$ bound, which serves as the primary pruning mechanism for the search tree.

## 1 Problem Definition

Given a set of $n$ items with integer sizes $I = \{s_1, s_2, \ldots, s_n\}$ and a fixed bin capacity $C$, the objective is to partition $I$ into the minimum number of subsets (bins) $B_1, \ldots, B_m$ such that the sum of sizes in each bin does not exceed $C$.

We assume without loss of generality that items are sorted in non-increasing order of size:

$$s_1 \geq s_2 \geq \cdots \geq s_n \tag{1}$$

## 2 Mathematical Lower Bounds

The efficiency of the MTP depends on the tightness of its lower bounds. A lower bound $L(I)$ allows the algorithm to prune branches of the search tree where the current partial solution plus the lower bound of the remaining items exceeds the best known solution.

### 2.1 The $L_1$ Bound (Volume Bound)

The simplest lower bound is derived from the continuous relaxation of the problem (assuming items can be split like liquid).

$$L_1 = \left\lceil \frac{\sum_{i=1}^{n} s_i}{C} \right\rceil \tag{2}$$

It is proven that the worst-case performance ratio of $L_1$ is $1/2$. While fast ($O(n)$), it is often loose for instances where items are large relative to $C$.

### 2.2 The $L_2$ Bound (Martello-Toth Bound)

The core mathematical contribution of the MTP is the $L_2$ bound, which improves upon $L_1$ by analyzing "wasted" space that is mathematically unavoidable.

### 2.2.1 Definitions

For any integer parameter $K$ such that $0 \leq K \leq C/2$, we classify the items into three sets:

$$N_1(K) = \{i \in I : s_i > C - K\} \qquad \text{(Large items)}$$
$$N_2(K) = \{i \in I : C - K \geq s_i > C/2\} \qquad \text{(Medium items)}$$
$$N_3(K) = \{i \in I : C/2 \geq s_i \geq K\} \qquad \text{(Small items)}$$

### 2.2.2 The $L(K)$ Function

Based on these sets, we define a function $L(K)$:

$$L(K) = |N_1| + |N_2| + \max\left(0, \left\lceil \frac{\sum_{i \in N_3} s_i - R_{N_2}}{C} \right\rceil\right) \tag{3}$$

where $R_{N_2}$ is the total residual capacity left in the bins containing items from $N_2$:

$$R_{N_2} = |N_2|C - \sum_{i \in N_2} s_i$$

**Theorem 1** (Correctness of $L(K)$)**.** *For any $K \in [0, C/2]$, $L(K)$ is a valid lower bound on the optimal number of bins $m$.*

*Proof.* Consider the packing requirements of the sets $N_1$, $N_2$, and $N_3$:

1. **Separation of Large Items:** Every item in $N_1$ has size $s_i > C - K$. Every item in $N_2$ has size $s_i > C/2$. Since $K \leq C/2$, all items in $N_1 \cup N_2$ have size strictly greater than $C/2$. Therefore, no two items from $N_1 \cup N_2$ can fit into the same bin. This implies that at least $|N_1| + |N_2|$ bins are required just to hold these items.

2. **Incompatibility of $N_1$ and $N_3$:** An item from $N_3$ has size $s_i \geq K$. An item from $N_1$ has size $s_j > C - K$. The sum $s_i + s_j > C$. Thus, no item from $N_3$ can be placed in a bin containing an item from $N_1$.

3. **Filling the Gaps:** The items in $N_3$ must be packed either into the remaining space of bins containing $N_2$ items, or into completely new bins. The total available space in the bins utilized by $N_2$ items is exactly $R_{N_2} = |N_2|C - \sum_{i \in N_2} s_i$.

4. **Calculation:** The total size of items in $N_3$ is $\sum_{i \in N_3} s_i$. The portion of this total size that *cannot* fit into the $N_2$ bins is:

$$\text{Excess} = \max\left(0, \sum_{i \in N_3} s_i - R_{N_2}\right)$$

   This excess volume must go into new bins. The minimum number of additional bins required for this excess is $\lceil \text{Excess}/C \rceil$.

5. **Conclusion:** The total bins required is the sum of bins for $N_1 \cup N_2$ plus the additional bins for the overflow of $N_3$.

$$m \geq (|N_1| + |N_2|) + \left\lceil \frac{\max(0, \sum_{N_3} s_i - R_{N_2})}{C} \right\rceil$$

This concludes the proof. □

#### 2.2.3 The Optimized $L_2$ Bound

The bound $L_2$ is defined as the maximum value of $L(K)$ over all feasible $K$:

$$L_2 = \max_{0 \le K \le C/2} L(K) \tag{4}$$

To compute this efficiently, it is sufficient to check $K$ only for distinct values of $s_i \le C/2$. If items are sorted, this can be computed in $O(n)$ time.

## 3 Reduction Procedures

Before and during the branching process, MTP applies reduction procedures to fix bins permanently, reducing the problem size. This relies on the concept of **Dominance**.

**Definition 1** (Feasible Set Dominance)**.** *A feasible set $F_1$ dominates a feasible set $F_2$ if the optimal solution obtained by setting a bin $B = F_1$ is not worse than the optimal solution obtained by setting $B = F_2$.*

### 3.1 Dominance Criterion

A sufficient condition for dominance is: If $F_1$ and $F_2$ are distinct feasible sets, and there exists a partition $P = \{P_1, \ldots, P_\ell\}$ of $F_2$ and a subset $\{i_1, \ldots, i_\ell\} \subseteq F_1$ such that:

$$s_{i_h} \ge \sum_{k \in P_h} s_k \quad \text{for } h = 1, \ldots, \ell \tag{5}$$

then $F_1$ dominates $F_2$.

### 3.2 Reduction Algorithm

The MTP reduction algorithm (Procedure REDUCTION) iteratively looks for dominating sets to fix into bins.

```
Initialize fixed bins count j := 0, Unpacked set I
repeat
    Let i be the largest remaining item.
    Find feasible set F containing i that dominates all other sets containing i.
    Check 1 (Single Item): If i cannot fit with any other item, F = {i}.
    Check 2 (Pairs): If i fits with i', and {i, i'} fills the bin so well (e.g., s_i + s_i' = C) that
  no triplet could be better, F = {i, i'}.
    if F ≠ ∅ then
        Fix Bin B_{j+1} := F.
        Remove items in F from I.
    end if
until No further reductions possible
```

This procedure essentially greedily matches items if they form a "perfect" or "dominant" bin, reducing the complexity for the subsequent Branch-and-Bound phase.

## 4 The Exact Branch-and-Bound Algorithm

The complete Martello-Toth Procedure combines the bounds and reductions into a Depth-First Search (DFS).

### 4.1 Algorithm Steps

1. **Initialization:**
    - Sort items $s_1 \geq s_2 \cdots \geq s_n$.
    - Calculate global lower bound $LB = L_2$.
    - Calculate heuristic upper bound $UB$ (using First-Fit Decreasing or Best-Fit Decreasing).
    - If $LB == UB$, the heuristic solution is optimal. STOP.
2. **Reduction:**
    - Apply Procedure REDUCTION to fix easy bins. Update problem size and $LB$.
3. **Branching (Backtracking):**
    - MTP builds a solution bin by bin.
    - For the current bin, it attempts to place the largest available item.
    - It then recursively attempts to fill the remaining capacity of the current bin with the next largest fitting items.
    - Once a bin is closed, it moves to the next bin.
4. **Pruning (The Critical Step):** At any node in the search tree:
    - Let $m_{current}$ be the number of bins already fixed/filled.
    - Let $I_{rem}$ be the set of remaining unpacked items.
    - Calculate the lower bound for the remainder: $L_2(I_{rem})$.
    - **Condition:** If $m_{current} + L_2(I_{rem}) \geq UB$, then this branch cannot lead to a better solution than what we already found. **PRUNE (Backtrack)**.
5. **Updating Best Solution:** If a valid packing is found with $z$ bins and $z < UB$:
    - Update $UB = z$.
    - If $UB == LB$, STOP (Optimal found).

## 5 Complexity and Optimality

The MTP is an exact algorithm. While the worst-case time complexity is exponential (due to the NP-hardness of BPP), the effective use of the $L_2$ bound allows it to solve many instances efficiently. The $L_2$ bound has an asymptotic worst-case performance ratio of 2/3, meaning it is tighter than simple volume bounds.