Bin packing – An approximation algorithm

How good is the FFD heuristic - A weak bound

Problem: We are concerned with storing/packing of objects of different sizes, with the objective of minimizing the amount of wasted space. The bin packing problem is posed formally as follows:

Let \( S = (s_1, \cdots, s_n) \), where \( 0 < s_i \leq 1 \) for \( i = 1, \cdots, n \) be the sizes of \( n \) given objects. It is required to find a partition \( U_1, \cdots, U_N \) of \( S \) where the sum of the sizes of the objects in each partition is at most 1 and \( N \) the minimum. It is possible to treat each partition as a bin of unit size. Then the problem is to pack the given objects into as few bins (call this value \( \text{opt}(S) \)) as possible.

It is convenient to refer to \( s_i \) as the corresponding object itself.

Applications: The problem arises packing files on disk tracks, program segments into memory pages, packing TV commercials into station breaks etc.

Worst case: \( \text{opt}(S) = n \) – this necessary and sufficient.

Brute-Force approach: Consider all ways to partition \( S \), so that total size of the objects in each partition is \( \leq 1 \). The number of possibilities exceed \( (\frac{n}{2})^\frac{n}{2} \). It is unlikely that the problem can be solved by a polynomial-time algorithm in view of the following result:
THEOREM: The bin packing problem is $NP$–hard.

The proof follows from a reduction of the subset-sum problem to bin packing.

THE FIRST-FIT DECREASING HEURISTIC (FFD)

- FFD is the traditional name – strictly, it is first-fit nonincreasing.
- An early known approximation algorithm.
- Works on greedy strategy.
- Produces good solutions in practice.
- Has a running time $O(n^2)$ in the worst case.

Algorithm FFD

1. Order the given objects in a non-decreasing order so that we have $s_1 \geq \cdots \geq s_n$. Initialize a counter $N = 0$.
2. Let the bins be $B_1, \cdots, B_n$. Put the next (first) object in the first “possible” bin, scanning the bins in the order $B_1, \cdots, B_n$.
   If a new bin is used, increment $N$.
3. Return $N$.

Complexity: Step 1 takes $\Theta(n \log n)$ time. In Step 2, the first object requires a scan of $B_1$ only. Second object requires scanning at most $B_1$ and $B_2$; etc. Therefore, the total number of scans is in $O(n^2)$, which is also the worst-case running time of FFD. It can be seen that FDD can be implemented to run in worst-time $\Theta(n \log n)$.
FFD is not optimal

Example: Instance given - 0.6, 0.6, 0.5, 0.4, 0.3, 0.2, 0.2, 0.2.

FFD will pack these as

\[[0.6|0.4], [0.6|0.3|], [0.5|0.2|0.2|], [0.2|].\]

That is \( N = 4 \) in \( FFD \). The optimal value \( \text{opt}(S) = 3 \).

It is easy to see that there are infinite instances that require \( N \) bins by \( FFD \) when \( \text{opt}(S) = N - 1 \).

The theorem below tells how bad can be \( FFD \) - that is, how bad is \( N \) in \( FFD \) as compared to \( \text{opt}(S) \). We begin with two lemmas.

**Lemma 1:** Let \( S = (s_1, \ldots, s_n) \) with \( s_1 \geq \cdots \geq s_n \). Let \( N > \text{opt}(S) \).
Let \( s \) be the size any object placed by \( FFD \) in any “extra” bin (selected from \( B_{\text{opt}(S)+1}, \ldots, B_N \)). Then \( s \leq \frac{1}{3} \).

**Proof:** Let \( s_i \) be the first object placed by \( FFD \) in \( B_{\text{opt}(S)+1} \). Since the objects in \( S \) are in nondecreasing order, it suffices to show \( s_i \leq \frac{1}{3} \).

Assume \( s_i > \frac{1}{3} \). This implies that, when \( s_i \) is picked by \( FFD \) \( s_1, \ldots, s_{i-1} > \frac{1}{3} \). Therefore the number of objects in each of the bins \( B_1, \ldots, B_{\text{opt}(S)} \) is \( \leq 2 \).

**Claim:** There exists a \( k \geq 0 \) such that the first \( k \) bins contain one object each and the remaining \( \text{opt}(S) - k \) bins contain two objects each.

If not, there will be two bins \( B_p \) and \( B_q \) where \( p < q \), as shown below

\[
B_p : [s_t | s_u|] \quad B_q : [s_v|],
\]

where \( B_p \) will have 2 objects and \( B_q \) will have only 1. As the objects are considered by \( FFD \) in nondecreasing order, \( s_t \geq s_v \) and \( s_u \geq s_i \). Hence
1 \geq s_t + s_u \geq s_u + s_i. This implies that \textit{FFD} would have placed \( s_i \) in \( B_q \) and the claim is true.

Therefore, the objects are filled thus:

\[ B_1 : [s_1] \, \cdots \, B_k : [s_k] \, \cdots \, B_{k+1} : [s_{k+1}|s_x|] \cdots. \]

Since \textit{FFD} did not place any of \( s_{k+1}, \cdots, s_i \) in the first \( k \) bins, none of these objects will fit together with any of \( s_1, \cdots, s_k \) in any bin. That is, in the optimal solution, the objects \( s_1, \cdots, s_k \) are the sole objects in their bins.

Without loss of generality, let these bins be the first \( k \) bins in the optimal solution; the remaining objects \( s_{k+1}, \cdots, s_{i-1} \) will be in bins \( B_{k+1}, \cdots, B_{\text{opt}(S)} \).

As the sizes of all these objects are \( > \frac{1}{3} \) and as these bins contain two objects, \( s_i \) cannot fit into any bin in the optimal solution. This is a contradiction and therefore the assumption \( s_i > \frac{1}{3} \) must be false.

\[ \text{Lemma 2: For any instance } S = (s_1, \cdots, s_n), \text{ the total number of objects placed by } \textit{FFD} \text{ in the extra bins is } \leq \text{opt}(S) - 1. \]

\[ \text{Proof: All the objects can be packed into } \text{opt}(S) \text{ bins. Therefore} \]

\[ \sum_{i=1}^{n} s_i \leq \text{opt}(S). \] (1)

Assume that \textit{FFD} places \text{opt}(S) objects with sizes \( t_1, \cdots, t_{\text{opt}(S)} \) in the extra bins.

Let \( b_j = \text{final contents (size) of bin } B_j, \text{ where } 1 \leq j \leq \text{opt}(S). \)

Now, \( b_j + t_j > 1 \) as otherwise \textit{FFD} could have placed \( t_j \) in \( B_j \).
Therefore

\[
\sum_{i=1}^{n} s_i \geq \sum_{j=1}^{opt(S)} b_j + \sum_{j=1}^{opt(S)} t_j = \sum_{j=1}^{opt(S)} (b_j + t_j) > opt(S).
\]  

(2)

Clearly, (2) contradicts (1). Therefore the assumption is wrong. ■

**Theorem:** For an input instance \( S \), let \( \rho_{FFD}(m) \) be defined as \( N/m \), where \( N \)=value returned by \( FFD \) and \( m = opt(S) \). Then

\[
\rho_{FFD}(m) \leq \frac{4}{3} + \frac{1}{3m}.
\]

(3)

**Proof:** \( FFD \) puts at most \( m - 1 \) objects into the extra bins. Each of those objects have a size \( \leq \frac{1}{3} \). Therefore

\[
N \leq m + \left\lceil \frac{(m-1)}{3} \right\rceil.
\]

Now, \( \rho_{FFD}(m) = \frac{N}{m} \leq 1 + \frac{m+1}{3m} \leq \frac{4}{3} + \frac{1}{3m} , \)

which establishes (3). ■

The known stronger result is: \( \rho_{FFD}(m) \leq \frac{11}{9} + \frac{4}{m} \). It has been emperically found that the expected number of extra bins used by \( FFD \) is about \( 0.3\sqrt{n} \).

**Reference**