Comp 7720 - Online Algorithms

Assignment 3: Bin Packing and Applications

Shahin Kamalli
University of Manitoba - Fall 2017

Due: Saturday, November 25th at 8:00pm (firm deadline)

November 28, 2017

‘The whole purpose of education is to turn mirrors into windows’ Sydney Harris

Please pay attention to the followings when preparing/submitting your assignment:

- All problems are written problems. There are five problems. Except for problem 4-part(b) which is bonus, all other problems are mandatory.

- If there is a result in the slides that can be used in your answers, you are welcome to refer to that.

- If you have any question related to the assignment, you are encouraged to post it on Piazza. Note that you can submit anonymously (in case you are shy). Do not post any answer or hint on Piazza (for the obvious reason).

- You are welcome to discuss the problems with your friends (or enemies). But you should write your answers individually.

- Submit your answers electronically using UMLearn. You should submit a single pdf file. You are encouraged to prepare your assignment using \LaTeX.

- The assignment is designed in a way to be simpler and shorter compared to other assignments so that you can spend more time on your projects.
Problem 1  Worst-Fit Algorithm for Bin Packing [15 marks]

Recall that Worst-Fit algorithm is an Any-Fit algorithm which places an incoming item into the bin with maximum left capacity. If such a bin does not exists, it opens a new bin for an incoming item.

a) Show the final packing of Worst-Fit for sequence \( \sigma = \langle 0.25, 0.8, 0.25, 0.15, 0.65, 0.85, 0.1, 0.3 \rangle \) (the intermediate packings are not required).

Answer:
Here is the final packing:

<table>
<thead>
<tr>
<th>0.1</th>
<th></th>
<th></th>
<th>0.3</th>
<th>0.85</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.25</td>
<td>0.8</td>
<td>0.65</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b) Provide a sequence that shows the competitive ratio of Worst-Fit is at least 2.

Answer:
Consider the following sequence, which is similar to the worst-case sequence for Next-Fit:

\[ \sigma = \langle 0.5, \epsilon, 0.5, \epsilon, 0.5, \epsilon, \ldots \rangle \]

Worst-fit places each item \( x \) of size \( \epsilon \) together with an item of size 0.5 that was opened just before \( x \). Hence, the level of the bin becomes a bit larger than half, and consequent items will not be placed in such bin.

The packing of Worst Fit will be:

<table>
<thead>
<tr>
<th>( \epsilon )</th>
<th>( \epsilon )</th>
<th>( \epsilon )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

while the packing of Opt is:

<table>
<thead>
<tr>
<th>0.5</th>
<th>0.5</th>
<th>( \epsilon )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>( \epsilon )</td>
</tr>
</tbody>
</table>

For \( n \) requests of the above sequence, Worst Fit opens roughly \( n/2 \) bins (i.e., number of items of size 0.5) while Opt opens roughly \( n/4 + 1 \) bins. So, we came up for one sequence for which the cost of Worst-Fit is 2 times that of Opt. Hence, the competitive ratio of Worst Fit (which is the maximum ratio among all sequences) is at least 2.

c) Show that the lower bound of part (b) is tight by showing the competitive ratio of Worst-Fit is at most 2.

Answer:
The proof for upper bound is similar to that of Next Fit. What the two algorithms have in common and makes their analysis similar is the fact that they both avoid opening a new bin if an incoming items fits in the last opened bin.
Consider any two consecutive bins in the final packing of Worst-Fit. We claim that the total size of items in these bins is more than 1. Consider otherwise, i.e., assume total size is less than or equal to one. This implies that all items in the second bin fit in the first bin. In particular, the first item which cause opening the second bin could fit in the first bin. However, this contradicts the fact that Worst-Fit opens a new bin only if an item does not fit in existing bins.

Let \( S \) denote the total size of items in an input sequence. Assume Worst-Fit opens \( k \) bins and let \( a_1, \ldots, a_k \) denote the total size of items in these \( k \) bins. By the above observation, items in any two consecutive bins have a total size of 1. Hence, we have \( a_1 + a_2 > 1; a_3 + a_4 \geq 1; \ldots; a_{k-1} + a_k \geq 1 \) (we assume \( k \) is even here; otherwise we ignore the last bins). So, there are \( k/2 \) pairs here and items in each pair have a total size of more than 1. Since the total size of all items is \( S \), the number of bins cannot be more than \( S/2 \) (otherwise, their total size will be more than \( S \)). In short, we have \( k < S/2 \) which implies \( \text{cost(WorstFit)} < 2S \). Let \( m \) denote the cost of Opt and let \( o_1, \ldots, o_m \) denote the level of items in bins of Opt. Note that we have \( o_i \leq 1 \) (total size of items in each bin is no more than capacity 1 of the bins). So, we have \( m \geq S \).

In short, we showed that for any sequence of total size \( S \), we have \( \text{cost(WorstFit)} < 2S \) and \( \text{cost(Opt)} \geq 2S \). We conclude that for any sequence, cost of Worst-Fit is within a ratio 2 of Opt, which implies that the competitive ratio of Worst-Fit is at most 2.
Problem 2  Upper & Lower bounds [15 marks]

Consider a restricted version of bin packing where all items larger than $1/3$.

a) Use a weighting argument to prove that the competitive ratio of First Fit is at most 1.5.

**Answer:**
Recall that in a weighting technique argument, in order to show an algorithm has a competitive ratio of at most $c$, we take the following steps:

– Step 1: we define a weight for all items. Here, we define the weight of an item with size larger than $1/2$ to be 1 and the weight of an item with size in the range $(1/3,1/2]$ to be $1/2$.

– Step 2: we show that, for any bin opened by the algorithm (here First Fit), the total the weight of all items is at least 1. Here, we consider two types of bins: those which include an item of size larger than $1/2$ have weight at least 1 (because that item has weight 1). Those bins which do not include an item of size larger than $1/2$ include exactly two items of size $(1/3,1/2]$ (with the possible exception of the last such bin). To see that, note that any two items in this range fit in the same bin and hence the algorithm will not open a new bin if an existing bin has only one item of size in $(1/3,1/2]$. Hence, all such bins include two items, each weighting $1/2$, which gives a total weight of 1. In short, we showed that total weight of items in any bin (except possibly one) is at least 1.

– Step 3: we show the total weight of items in a bin $B$ of OPT is at most $c = 1.5$. Again we do a case analysis: If $B$ includes an item of size larger than $1/2$ (with weight 1), it has space for at most one other item of size smaller than $1/2$ (with weight $1/2$), which gives a total weight of 1.5. If $B$ does not include an item larger than $1/2$, then it can include at most two items of weight $1/2$ each, which results in total weight of 1. Hence, total weight of items in $B$ is no more than 1.5.

b) Provide a lower bound for competitive ratio of any algorithm for these instances.

**Answer:**
In Lecture.15, Slide. 14-16, we considered the following sequence:

$$\sigma = 1/2 - \epsilon, 1/2 - \epsilon, \ldots, 1/2 - \epsilon, 1/2, 1/2 + \epsilon, \ldots, 1/2 + \epsilon$$

There, we proved that the competitive ratio of any algorithm for this input sequence is at least $4/3$. Note that all items in the above sequence are larger than $1/3$. We conclude that, even for sequences where all items are larger than $1/3$, no online algorithm can have a competitive ratio better than $4/3$. 


Problem 3  Harmonic algorithm [15 marks]

Consider the input sequence \( \sigma = \langle 0.2, 0.3, 0.4, 0.35, 0.8, 0.12, 0.6, 0.2, 0.1, 0.33, 0.05 \rangle \). In what follows, you are asked to provide final packings of a few bin packing algorithms for packing \( \sigma \). In doing so, showing intermediate packings is not required.

a) Show the final packing of Harmonic with parameter \( k = 3 \) for \( \sigma \).

**Answer:**

Here it is:

\[
\begin{array}{ccc}
\text{class1} & \text{class2} & \text{class3} \\
0.8 & 0.6 & 0.35 \\
0.4 & 0.2 & 0.3 \\
0.3 & 0.2 & 0.05 \\
0.1 & 0.12 & 0.33
\end{array}
\]

**Note:** In most implementations of Harmonic algorithm, items of class \( k \) are placed using Next Fit strategy; so, when a new bin is opened for item with size 0.33, the previous bin is closed and the next item is placed together with 0.33. However, you will get the full mark if you assume First Fit or any other Any-Fit strategy for items of class \( k \).

b) Show the final packing of Harmonic with parameter \( k = 5 \) for \( \sigma \).

**Answer:**

Here it is (note that there is no bin of class 4 because there is no item in the range \([1/3,1/4)\)):

\[
\begin{array}{cccc}
\text{class1} & \text{class2} & \text{class3} & \text{class5} \\
0.8 & 0.6 & 0.35 & 0.4 \\
0.33 & 0.2 & 0.05 & 0.2 \\
0.12 & 0.2
\end{array}
\]

c) Show the final packing of Best Fit for \( \sigma \).

**Answer:**

Here it is:

\[
\begin{array}{cccc}
\text{class1} & \text{class2} & \text{class3} & \text{class4} \\
0.1 & 0.4 & 0.05 & 0.6 \\
0.3 & 0.6 & 0.12 & 0.35 \\
0.2 & 0.8 & 0.33 & 0.2
\end{array}
\]
Problem 4  Any Fit algorithm [10 + 10 marks]

Consider a restriction of bin packing where all items are smaller than 1/k.

a) Prove that competitive ratio of any Any-Fit algorithm is at most \( \frac{k}{k-1} \).

Answer:

The level of any bin \( B \) (except the last opened bin) is at least \( 1 - \frac{1}{k} \). Otherwise, the bins opened after \( B \) are opened by an item of size larger than \( 1/k \) (contradiction). So, if the total size of all items is \( S \), we have \( \text{Alg}(\sigma) \leq \frac{S}{1-1/k} \). On the other hand, since items in each bin cannot have total size more than 1, we have \( \text{Opt}(\sigma) \geq S \). So, for any sequence of total size \( S \), the ratio between \( \text{Alg}(\sigma) \) and \( \text{Opt}(\sigma) \) is at most \( k/k-1 \), i.e., the competitive ratio is at most \( \frac{k}{k-1} \).

b) [Bonus] Prove that First Fit has a better competitive ratio of at most \( \frac{k+1}{k} \).

Answer:

We show that all bins except at most 2 have level at least \( 1 - 1/(k+1) = k/(k+1) \). Consider the first bin \( B \) with level less than \( k/(k+1) \). That means all items placed in the bins opened after \( B \) only include items in the range \( [1/(k+1), 1/k) \). Any bin opened after \( B \) has space for any set of \( k \) such items. Let \( B' \) be any such bin. Before opening any bin after \( B' \), \( k \) items are placed in \( B' \) and hence it has a level of at least \( k/(k+1) \). We conclude that all bins, except possibly two, have level \( k/(k+1) \).

The rest of the proof is similar to part(a): if the total size of all items is \( S \), we have \( \text{Alg}(\sigma) \leq \frac{S}{k/(k+1)} \). On the other hand, since items in each bin cannot have total size more than 1, we have \( \text{Opt}(\sigma) \geq S \). So, for any sequence of total size \( S \), the ratio between the cost of an Ant-Fit algorithm and \( \text{Opt} \) is at most \( \frac{k+1}{k} \), i.e., the competitive ratio is at most \( (k+1)/k \).
Problem 5  Fault-tolerant bin packing [10 marks]

Consider the following two packings for an instance of fault-tolerant bin packing. Indicate whether each of them is valid or invalid. Justify your answer.

Answer:

Packing (a) is valid: if $S_1$ fails, the extra load to $S_2$ increases its level from 0.6 to 0.7 which is within its capacity. Similarly, the load of $S_3$ is increased from 0.7 to 0.8 which is fine. Similarly, if $S_2$ fails, level of $S_1$ and $S_3$ is increased to 0.9 (for both), which is fine. If $S_3$ fails, the loads of $S_1 \ldots S_5$ becomes 0.6, 0.8, - , 0.8, 0.8, which are all less than 1. If $S_4$ fails, levels of $S_3$ and $S_5$ becomes respectively 0.9 and 1.0 which is fine. Finally, if $S_5$ fails, levels of $S_3$ and $S_4$ becomes 0.9 and 1.0, which is within the capacity.

Packing (b) is invalid: if $S_3$ fails, the load for replica of size 0.4 redirects to $S_4$. At this point, the total load of $S_4$ would be 0.4 + 0.25 + 0.4 which is larger than capacity 1 of the servers and an overflow happens.