Online Bin Packing

Shahin Kamali

Lecture 20 - Nov. 20th, 2018

University of Manitoba
Review & Plan
Today’s objectives

- Online bin packing with advice
  - How many bits of advice is sufficient to achieve an optimal solution?
  - How many bits of advice is sufficient to break the lower bound for competitive ratio of any online algorithm?
Under the advice model, an online algorithm receives $b$ bits of advice from an benevolent offline oracle.

The advice bits are available since the beginning.

There is a compromise between the number of advice bits ($b$) and quality of algorithms (e.g., their competitive ratio).
For a fixed sequence of fixed length $n$:

- How many bits of advice are required (sufficient) to achieve an optimal solution?
- How many bits of advice are sufficient to outperform all online algorithms?
- How good the competitive ratio can be with an advice of linear/sublinear size?
Theorem

For any sequence of length $n$, advice of size $O(n \log k)$ is sufficient to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

What advice encodes?
Optimal Solution with Advice

Theorem

For any sequence of length $n$, advice of size $O(n \log k)$ is sufficient to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

- What advice encodes?
  - For each item, it encodes the bin that it is packed to in an optimal packing.
Theorem

For any sequence of length \( n \), advice of size \( O(n \log k) \) is sufficient to achieve an optimal solution, where \( k \) is number of bins in an optimal packing.

- What advice encodes?
  - For each item, it encodes the bin that it is packed to in an optimal packing

- What is the advice size?
  - For each item, we require \( O(\log k) \) bits to encode the target bin. In total, \( O(n \log k) \) bits suffice.
Optimal Solution with Advice

Theorem

For any sequence of length $n$, advice of size $O(n \log k)$ is sufficient to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

- What advice encodes?
  - For each item, it encodes the bin that it is packed to in an optimal packing

- What is the advice size?
  - For each item, we require $O(\log k)$ bits to encode the target bin. In total, $O(n \log k)$ bits suffice.
Optimal Solution with Advice

Theorem

For any sequence of length $n$, advice of size $O(n \log k)$ is sufficient to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

- What advice encodes?
  - For each item, it encodes the bin that it is packed to in an optimal packing

- What is the advice size?
  - For each item, we require $O(\log k)$ bits to encode the target bin. In total, $O(n \log k)$ bits suffice.

- How the algorithm work with the given advice?
Theorem

For any sequence of length $n$, advice of size $O(n \log k)$ is sufficient to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

- What advice encodes?
  - For each item, it encodes the bin that it is packed to in an optimal packing

- What is the advice size?
  - For each item, we require $O(\log k)$ bits to encode the target bin. In total, $O(n \log k)$ bits suffice.

- How the algorithm work with the given advice?
  - It packs each item in the same bin as $\text{OPT}$ does.
Theorem

For any sequence of length \( n \), advice of size \( O(n \log k) \) is sufficient to achieve an optimal solution, where \( k \) is number of bins in an optimal packing.

- What advice encodes?
  - For each item, it encodes the bin that it is packed to in an optimal packing.

- What is the advice size?
  - For each item, we require \( O(\log k) \) bits to encode the target bin. In total, \( O(n \log k) \) bits suffice.

- How the algorithm work with the given advice?
  - It packs each item in the same bin as \( \text{OPT} \) does.

- Why it is optimal? The resulting packing is similar to \( \text{OPT} \).
**Theorem**

For any sequence of length $n$, $\Omega(n \log k)$ bits of advice are required to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

$$\sigma = \langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \rangle$$
Optimal Solution with Advice

Theorem

For any sequence of length $n$, $\Omega(n \log k)$ bits of advice are required to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

$$\sigma = \langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \rangle$$

- Each of the first $n-k$ items can be packed in any of the $k$ bins
- The summation of all of them is less than $1/2$. 

Review & Plan

COMP 7720 - Online Algorithms
Online Bin Packing
**Theorem**

For any sequence of length $n$, $\Omega(n \log k)$ bits of advice are required to achieve an optimal solution, where $k$ is number of bins in an optimal packing.

$$\sigma = \left\langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \right\rangle$$

- Each of the first $n - k$ items can be packed in any of the $k$ bins.
- The summation of all of them is less than 1/2.
- There will be $\frac{k^{n-k}}{k!}$ different packings!
Optimal Solution with Advice

$$\sigma = \left\langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \right\rangle$$

There will be \( \frac{k^{n-k}}{k!} \) different packings!
Review & Plan

Optimal Solution with Advice

\[ \sigma = \left\langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \right\rangle \]

- There will be \( \frac{k^{n-k}}{k!} \) different packings!
- For each packing the last \( k \) items (\( u_i \)'s) fill the empty space for each bin
Optimal Solution with Advice

\[ \sigma = \langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \rangle \]

- There will be \( \frac{k^{n-k}}{k!} \) different packings!
- For each packing the last \( k \) items (\( u_i \)'s) fill the empty space for each bin
  - \( \sigma_1 = \langle 1/4, 1/8, 1/16, 1 - 1/4, 1 - 1/8, 1 - 1/16 \rangle \)
  - \( \sigma_2 = \langle 1/4, 1/8, 1/16, 1 - 1/4 - 1/16, 1 - 1/8, 1 - 1/16 \rangle \)
  - \( \sigma_3 = \langle 1/4, 1/8, 1/16, 1 - 1/4 - 1/8, 1 - 1/16, 1 \rangle \)
  - \( \sigma_4 = \langle 1/4, 1/8, 1/16, 1 - 1/4 - 1/8 - 1/16, 1, 1 \rangle \)
Optimal Solution with Advice

\[ \sigma = \left\langle \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \ldots, \frac{1}{2^{n-k+1}}, u_1, u_2, \ldots, u_k \right\rangle \]

- There will be \(\frac{k^{n-k}}{k!}\) different sequences!
  - All start with the same prefix of length \(n - k\)

- For each sequence, an optimal algorithm should pack the first \(n - k\) items differently from others.
  - Each sequence requires an advice tailored for itself

- \(\frac{k^{n-k}}{k!}\) different advice strings are required
  - \(\log \frac{k^{n-k}}{k!} \approx (n - 2k) \log k\) advice bits are required.
To achieve an optimal packing, it is sufficient to receive \( n \lceil \log \text{Opt}(\sigma) \rceil \) bits of advice. Moreover, any deterministic online algorithm requires at least \((n - 2 \text{Opt}(\sigma)) \log \text{Opt}(\sigma)\) bits of advice to achieve an optimal packing.
Assume the sequence is formed by \( m = o(n) \) distinct items which have size larger than a fixed value \( \epsilon \).
Assume the sequence is formed by $m = o(n)$ distinct items which have size larger than a fixed value $\epsilon$.

**Theorem**

*It is sufficient to read $O(m \log n)$ bits of advice to achieve an optimal packing.*
Optimal Solution with Advice

- Assume the sequence is formed by $m = o(n)$ distinct items which have size larger than a fixed value $\epsilon$.

**Theorem**

*It is sufficient to read $O(m \log n)$ bits of advice to achieve an optimal packing.*

- For each item $x$ encode its frequency in the input sequence!
Optimal Solution with Advice

- Assume the sequence is formed by \( m = o(n) \) distinct items which have size larger than a fixed value \( \epsilon \).

Theorem

_It is sufficient to read \( O(m \log n) \) bits of advice to achieve an optimal packing._

- For each item \( x \) encode its frequency in the input sequence!
  - This requires \( O(\log n) \) bits
Assume the sequence is formed by $m = o(n)$ distinct items which have size larger than a fixed value $\epsilon$.

**Theorem**

*It is sufficient to read $O(m \log n)$ bits of advice to achieve an optimal packing.*

- For each item $x$ encode its frequency in the input sequence!
  - This requires $O(\log n)$ bits
- The advice encodes the whole multi-set that forms the input in $O(m \log n)$ bits.
- Given the multi-set, pack it, optimally, using an offline algorithm before starting to serve the input.
- When an item is revealed, place it into its reserved space in the offline packing!
The Idea Behind the Lower Bound

**Theorem**

To achieve an optimal solution least $\Omega(m \log n)$ bits of advice are required.

- Consider a subclass of sequences which start by $n/2$ items of size $\epsilon$. 

The Idea Behind the Lower Bound

**Theorem**

To achieve an optimal solution least $\Omega(m \log n)$ bits of advice are required.

- Consider a subclass of sequences which start by $n/2$ items of size $\epsilon$.
  - Let $X$ denote the number of ways that these $n/2$ items can be packed
    - $X$ will be at least $(1 + \frac{n}{(m-1)(m-2)})^{m-3}$ (we skip the proof here)
**The Idea Behind the Lower Bound**

- For each partial packing, complete the sequence with items which fill the empty spaces

Example: 
- $n=30$, $m=6$ (bin capacities scaled up by 12)
- sequence: $<1^{(15)}$ ...
The Idea Behind the Lower Bound

- For each partial packing, complete the sequence with items which fill the empty spaces.
- Each sequence requires a distinct advice, and consequently an advice of size $\lg X$ is required.

![$\log(1 + \frac{n}{(m-1)(m-2)})^{m-3} = m\log n + o(\log n)$ bits are required](image)

Example:

$n = 30$, $m = 6$ (bin capacities scaled up by 12)

Sequence: $<1^{(15)} 11 11 11 10 10 9 8 12^{(7)}> \ldots$

Sequence: $<1^{(15)} 11 11 11 11 11 11 8 8 12^{(6)}> \ldots$
It is sufficient to read \( O(m \log n) \) bits of advice to achieve an optimal packing. At least \( \Omega(m \log n) \) bits of advice are required to achieve an optimal solution.
Breaking the Lower Bound

- How many bits of advice are sufficient to achieve a competitive ratio better than all online algorithms?
  - We have seen before that no online algorithm can have a competitive ratio better than 1.54.
How many bits of advice are sufficient to achieve a competitive ratio better than all online algorithms?

- We have seen before that no online algorithm can have a competitive ratio better than 1.54.

- $O(\log n)$ bits of advice is sufficient to achieve competitive ratio 1.5.
How many bits of advice are sufficient to achieve a competitive ratio better than all online algorithms?

- We have seen before that no online algorithm can have a competitive ratio better than 1.54.

- $O(\log n)$ bits of advice is sufficient to achieve competitive ratio 1.5.

Divide items into four groups based on their sizes:

- **Huge** items: size larger than $2/3$
- **critical** items: size in range $(1/2, 2/3]$ 
- **mini** item: size in range $(1/3, 1/2]$ 
- **tiny** items: size smaller than $1/3$
Review & Plan

Breaking the Lower Bound

- Receive number of critical items with $O(\log n)$ bits of advice
- Consider **ReserveCritical** algorithm:
  - At the beginning, reserve a space of size 2/3 for critical items
  - **Huge** items: open a new bin
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin
  - **tiny** items: apply First-Fit to place in bins with critical or other tiny items
ReserveCritical Algorithm

At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle \]
ReserveCritical Algorithm

- At the beginning, reserve a space of size 2/3 for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \rangle \]
Review & Plan

ReserveCritical Algorithm

- At the beginning, reserve a space of size $2/3$ for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

$$\sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \rangle$$

![Diagram of bins with reserve space]
At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle \]
ReserveCritical Algorithm

At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \rangle \]
Review & Plan

ReserveCritical Algorithm

- At the beginning, reserve a space of size 2/3 for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

$$\sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \rangle$$
At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3, 0.9, 0.6, 0.5, 0.1, 0.1, 0.56, 0.4, 0.3, 0.45, 0.8, 0.51, 0.41, 0.2, 0.1, 0.37, 0.3 \rangle \]
ReserveCritical Algorithm

At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle \]
Reserve Critical Algorithm

At the beginning, reserve a space of size 2/3 for critical items

- *Huge* items: open a new bin (no other item goes there)
- *critical* items: place in a reserve space
- *mini* item: place two of them in the same bin (no other item goes there)
- *tiny* items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \left\langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \right\rangle \]
At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **Critical** items: place in a reserve space
- **Mini** item: place two of them in the same bin (no other item goes there)
- **Tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle \]
ReserveCritical Algorithm

At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

$$\sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \rangle$$
Reserve Critical Algorithm

At the beginning, reserve a space of size $2/3$ for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

$$\sigma = \langle 0.3, 0.9, 0.6, 0.5, 0.1, 0.1, 0.56, 0.4, 0.3, 0.45, 0.8, 0.51, 0.41, 0.2, 0.1, 0.37, 0.3 \rangle$$
ReserveCritical Algorithm

- At the beginning, reserve a space of size $2/3$ for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

\[
\sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle
\]
At the beginning, reserve a space of size $2/3$ for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

$$\sigma = \langle 0.3, 0.9, 0.6, 0.5, 0.1, 0.1, 0.56, 0.4, 0.3, 0.45, 0.8, 0.51, 0.41, 0.2, 0.1, 0.37, 0.3 \rangle$$
ReserveCritical Algorithm

- At the beginning, reserve a space of size \( \frac{2}{3} \) for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

\[
\sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \rangle
\]
At the beginning, reserve a space of size 2/3 for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle \]
At the beginning, reserve a space of size $2/3$ for critical items

- **Huge** items: open a new bin (no other item goes there)
- **critical** items: place in a reserve space
- **mini** item: place two of them in the same bin (no other item goes there)
- **tiny** items: apply FF to place in bins with critical or other tiny items

$$\sigma = \langle 0.3 \quad 0.9 \quad 0.6 \quad 0.5 \quad 0.1 \quad 0.1 \quad 0.56 \quad 0.4 \quad 0.3 \quad 0.45 \quad 0.8 \quad 0.51 \quad 0.41 \quad 0.2 \quad 0.1 \quad 0.37 \quad 0.3 \rangle$$
ReserveCritical Algorithm

- At the beginning, reserve a space of size 2/3 for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3, 0.9, 0.6, 0.5, 0.1, 0.1, 0.56, 0.4, 0.3, 0.45, 0.8, 0.51, 0.41, 0.2, 0.1, 0.37, 0.3 \rangle \]
Review & Plan

Reserve Critical Algorithm

- At the beginning, reserve a space of size 2/3 for critical items
  - **Huge** items: open a new bin (no other item goes there)
  - **critical** items: place in a reserve space
  - **mini** item: place two of them in the same bin (no other item goes there)
  - **tiny** items: apply FF to place in bins with critical or other tiny items

\[ \sigma = \langle 0.3 \ 0.9 \ 0.6 \ 0.5 \ 0.1 \ 0.1 \ 0.56 \ 0.4 \ 0.3 \ 0.45 \ 0.8 \ 0.51 \ 0.41 \ 0.2 \ 0.1 \ 0.37 \ 0.3 \ \rangle \]
ReserveCritical Analysis

- ReserveCritical has c.r. of at most 1.5.
- Based on the final packing, we consider two cases:
  - Case 1: There is a bin opened by a tiny item
  - Case 2: No bin is opened by a tiny item
Case 1: There is a bin opened by a tiny item

All bins (except possibly three) have a level of at least 2/3

We verify it for all bin types:
Case 1: There is a bin opened by a tiny item

All bins (except possibly three) have a level of at least 2/3

- We verify it for all bin types:
- **Huge bins**: all bin include one item of size larger than 2/3
Case 1: There is a bin opened by a tiny item

All bins (except possibly three) have a level of at least $2/3$

- We verify it for all bin types:
- **Huge bins:** all bin include one item of size larger than $2/3$
- **Mini-bins:** all bins (except the last one) include two items of size in range $(1/3,1/2]$
Case 1: There is a bin opened by a tiny item  
All bins (except possibly three) have a level of at least $2/3$  

- **Critical bins:** in all bins the non-reserved space (of size $1/3$) is filled with items of total size at least $1/6$; the level will be at least $1/2 + 1/6 = 2/3$. 

![Diagram of bins with sizes]
Case 1: There is a bin opened by a tiny item

All bins (except possibly three) have a level of at least 2/3

Critical bins: in all bins the non-reserved space (of size 1/3) is filled with items of total size at least 1/6; the level will be at least $1/2 + 1/6 = 2/3$.

- consider the first bin with less than 1/6 in the non-reserved space
- all tiny items placed in the following critical bins have size in the range $(1/6, 1/3]$
- exactly one of them fits in the reserved spaced of any following bin
- since there is a bin opened by tiny items, all following bins receive one tiny item in the range $(1/6, 1/3]$. 
Case 1: There is a bin opened by a tiny item

- All bins (except possibly three) have a level of at least 2/3
  - **Critical bins**: in all bins the non-reserved space (of size 1/3) is filled with items of total size at least 1/6; the level will be at least \(1/2 + 1/6 = 2/3\).
    - The choice of 1/6 is not arbitrary; if we made the claim for a smaller value (e.g., 1/7) the levels were less than 2/3
    - If we made the claim for a larger value (e.g., \(x = 1/5\)), then following the analysis, the tiny items placed after the first bin with level less than \(x\) in the reserved space will have size \(\geq 1/3 - x\), which is less than 1/6, and the level of critical bins, which receive exactly one such tiny item, will be \(1/2 + 1/3 - x\) which is less than 2/3.
Case 1: There is a bin opened by a tiny item

All bins (except possibly three) have a level of at least 2/3

Tiny bins: bins receive items of size less than 1/3

All bins (except the last one) will have level at least 2/3
Case 1: There is a bin opened by a tiny item

All bins (except possibly three) have a level of at least 2/3

Let $S$ be the total size of all items

- $\text{cost(ReserveCritical)} \leq S/(2/3) = 3S/2$
- $\text{cost(OPT)} \geq S$

The competitive ratio will be at most $\frac{3W/2}{W} = 3/2$
Case 2: No bin is opened by a tiny item $\rightarrow$ Use a weighting function
Case 2: No bin is opened by a tiny item → Use a weighting function

Step I: define a weighting function

- For any huge or critical item $x$, define $w(x) = 1$.
- For any mini item $y$, define $f(y) = 1/2$.
- For any tiny item $z$, define $f(z) = 0$. 
Case 2: No bin is opened by a tiny item → Use a weighting function

Step II: show for any bin in ReserveCritical packing, total weight of items in $B$ is at least 1:

- Huge bins: there is one huge item of weight 1

Diagram: 

- 0.3
- 0.6
- 0.1
- 0.1
- 0.3
- 0.56
- 0.51
- 0.9
- 0.4
- 0.41
- 0.45
- 0.8
Review & Plan

ReserveCritical Algorithm

- Case 2: No bin is opened by a tiny item → Use a weighting function

- Step II: show for any bin in ReserveCritical packing, total weight of items in $B$ is at least 1:
  - Huge bins: there is one huge item of weight 1
  - Critical bins: there is one critical item of weight 1
Case 2: No bin is opened by a tiny item → Use a weighting function

Step II: show for any bin in ReserveCritical packing, total weight of items in $B$ is at least 1:

- Huge bins: there is one huge item of weight 1
- Critical bins: there is one critical item of weight 1
- Mini bins: there are two items of weight $1/2$ each (except one bin)
Review & Plan

ReserveCritical Algorithm

- Case 2: No bin is opened by a tiny item → Use a weighting function.

- Step II: show for any bin in ReserveCritical packing, total weight of items in $B$ is at least 1:
  - Huge bins: there is one huge item of weight 1
  - Critical bins: there is one critical item of weight 1
  - Mini bins: there are two items of weight 1/2 each (except one bin)
  - Tiny bins: there is no tiny bin in case 2!
Review & Plan

ReserveCritical Algorithm

Case 2: No bin is opened by a tiny item → Use a weighting function

Step III: show that any bin $B$ in the packing of $OPT$ has total weight at most 1.5

- If $B$ includes a huge item the it cannot include any other item of non-zero weight → its weight will be 1

```
0.3  0.56  0.3  0.9  0.4  0.41  0.8
```
ReserveCritical Algorithm

Case 2: No bin is opened by a tiny item → Use a weighting function

Step III: show that any bin $B$ in the packing of $\text{OPT}$ has total weight at most 1.5

- If $B$ includes a huge item the it cannot include any other item of non-zero weight → its weight will be 1
- If $B$ includes a critical item, then it has space for at most one mini item → the weight of $B$ will be $1 + 1/2 = 1.5$. 
Review & Plan

ReserveCritical Algorithm

- Case 2: No bin is opened by a tiny item $\rightarrow$ Use a weighting function

- Step III: show that any bin $B$ in the packing of $\text{OPT}$ has total weight at most 1.5
  - If $B$ includes a huge item, it cannot include any other item of non-zero weight $\rightarrow$ its weight will be 1
  - If $B$ includes a critical item, then it has space for at most one mini item $\rightarrow$ the weight of $B$ will be $1 + 1/2 = 1.5$.
  - If $B$ includes no huge or critical item, it can include at most 2 mini items $\rightarrow$ the weight of $B$ will be $1/2 + 1/2 = 1$.

![Diagram of bins with weights]
Theorem

*Competitive ratio of ReserveCritical is at most 1.5.*

- With $O(\log n)$ bits of advice, one can achieve a competitive ratio of 1.5.
- Can we improve this? We will see the answer in the next class!