First Fit and Best Fit bin packing: A new analysis

Jiří Sgall
(joint work with György Dósa)

Computer Science Institute of the Charles Univ., Praha

Szeged, September 2012

A Brief Review of Bin Packing

Bin packing

- Input: Sequence of **items** $a_1, \ldots, a_n \in [0, 1]$.
- Output: Assign into **bins** of size 1.
- Objective: Minimize the number of bins.

A Brief Review of Bin Packing

Bin packing

- Input: Sequence of **items** $a_1, \ldots, a_n \in [0, 1]$.
- Output: Assign into **bins** of size 1.
- Objective: Minimize the number of bins.

Complexity results

- It is NP-hard to decide if $OPT(I) = 2$.
  Thus it is NP-hard to approximate with ratio $< 3/2$.
- There exists an asymptotic approximation scheme.
  I.e., in polynomial time we can pack the items into $(1 + \varepsilon)OPT(I) + 1$ bins.

Performance measures

Absolute approximation ratio

For each instance I, the algorithm gives

$$ALG(I) \leq R \cdot OPT(I)$$

Performance measures

Absolute approximation ratio

For each instance I, the algorithm gives

$$ALG(I) \leq R \cdot OPT(I)$$

Asymptotic approximation ratio

There exists a constant C such that for each instance I, the algorithm gives

$$ALG(I) \leq R \cdot OPT(I) + C$$

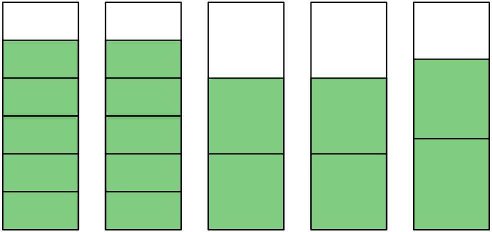
Two online algorithms

First Fit

Packs items one by one, always into the **first** bin where it fits.
Opens a new bin only when necessary.

Best Fit

Packs items one by one, always into the **most full** bin where it fits.
Opens a new bin only when necessary.



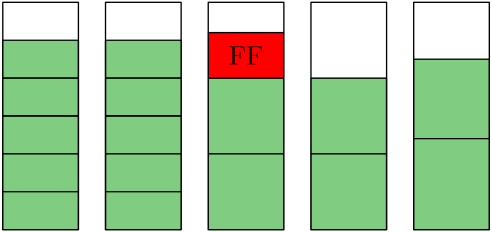
Two online algorithms

First Fit

Packs items one by one, always into the **first** bin where it fits.
Opens a new bin only when necessary.

Best Fit

Packs items one by one, always into the **most full** bin where it fits.
Opens a new bin only when necessary.



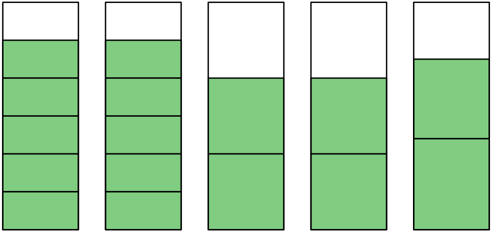
Two online algorithms

First Fit

Packs items one by one, always into the **first** bin where it fits.
Opens a new bin only when necessary.

Best Fit

Packs items one by one, always into the **most full** bin where it fits.
Opens a new bin only when necessary.



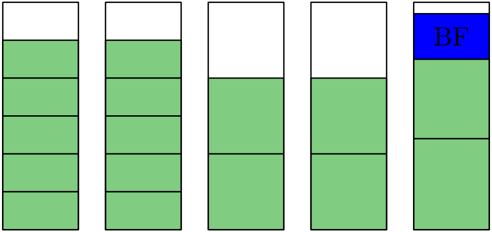
Two online algorithms

First Fit

Packs items one by one, always into the **first** bin where it fits.
Opens a new bin only when necessary.

Best Fit

Packs items one by one, always into the **most full** bin where it fits.
Opens a new bin only when necessary.



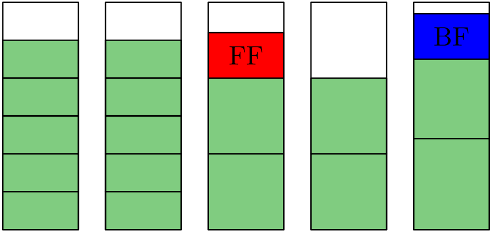
Two online algorithms

First Fit

Packs items one by one, always into the **first** bin where it fits.
Opens a new bin only when necessary.

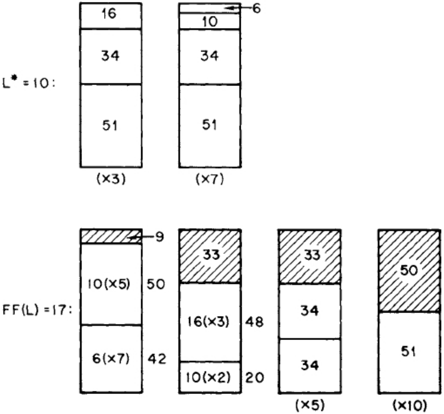
Best Fit

Packs items one by one, always into the **most full** bin where it fits.
Opens a new bin only when necessary.



Known Results

The asymptotic approximation ratio of both First Fit and Best Fit is equal to 1.7. More precisely, $FF, BF \leq \lceil 1.7 \cdot OPT \rceil$.
Example with $FF = BF = 17$ and $OPT = 10$:



Bin size is 101.

Overview of this talk

1. An easy proof of the 1.7 asymptotic approximation ratio for First Fit.
2. An extension for Best Fit.
3. A proof of the 1.7 **absolute** approximation ratio for First Fit.

Overview of this talk

1. An easy proof of the 1.7 asymptotic approximation ratio for First Fit.
2. An extension for Best Fit.
3. A proof of the 1.7 **absolute** approximation ratio for First Fit.

Main Technique

Classical technigue: **Weight functions**.
Find a weight of items such that

- Each bin in OPT has weight ≤ 1.7.
- Each bin in FF (BF) has weight ≥ 1 on average.

Overview of this talk

1. An easy proof of the 1.7 asymptotic approximation ratio for First Fit.
2. An extension for Best Fit.
3. A proof of the 1.7 **absolute** approximation ratio for First Fit.

Main Technique

Classical technigue: **Weight functions**.

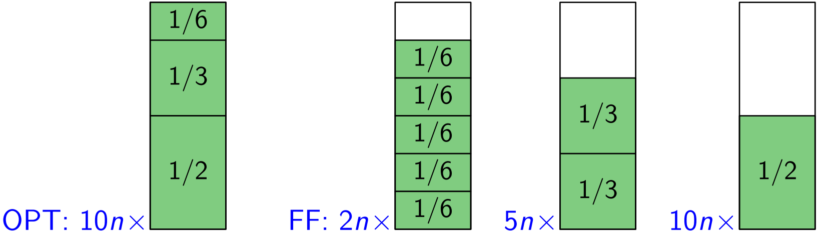
Find a weight of items such that

- Each bin in OPT has weight ≤ 1.7.
- Each bin in FF (BF) has weight ≥ 1 on average.

Combine weight functions with **amortized analysis**.

Idealized example for First Fit

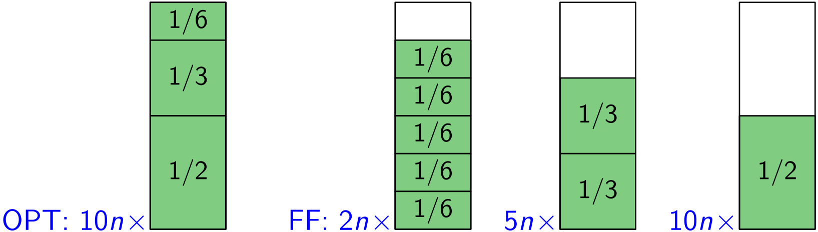
Assume that the algorithm cannot have bins of size exactly 1.



Essentially, this can be achieved by changing the item sizes by a small amount.

Idealized example for First Fit

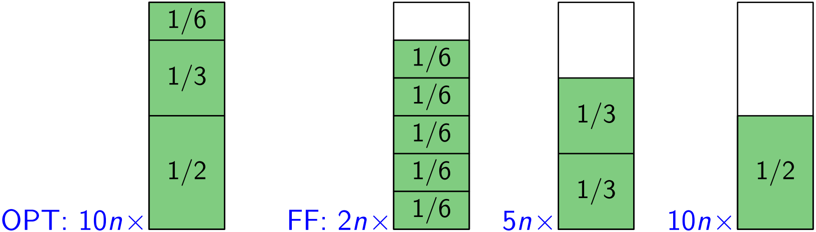
Assume that the algorithm cannot have bins of size exactly 1.



What should be the weights?

Idealized example for First Fit

Assume that the algorithm cannot have bins of size exactly 1.



$w(1/6) = 0.2 \quad w(1/3) = 0.5 \quad w(1/2) = 1$

The weight function

- Weight: **Scaled size** plus a **bonus.**
- $w(a) = \frac{6}{5}a + b(a)$

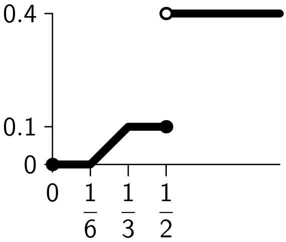
The weight function

- Weight: **Scaled size** plus a **bonus.**

- $w(a) = \frac{6}{5}a + b(a)$

- $b(a) =$

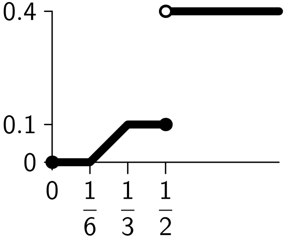
$$\begin{cases} 0 & \text{if } a \leq \frac{1}{6} \\ \frac{3}{5}\left(a - \frac{1}{6}\right) & \text{if } a \in [\frac{1}{6}, \frac{1}{3}] \\ 0.1 & \text{if } a \in [\frac{1}{3}, \frac{1}{2}) \\ 0.4 & \text{if } a > \frac{1}{2} \end{cases}$$



Offline bins

Each bin (a set of items of size ≤ 1) contains **bonus items**:

- either no item of size $> 1/2$ and at most 5 items with bonus at most 0.1 each (actually the total is < 0.3),
- or one item of size $> 1/2$ and at most 2 items with bonus at most 0.1 total.



Offline bins

Each bin (a set of items of size ≤ 1) contains **bonus items**:

- either no item of size $> 1/2$ and at most 5 items with bonus at most 0.1 each (actually the total is < 0.3),
- or one item of size $> 1/2$ and at most 2 items with bonus at most 0.1 total.

- Thus the **total bonus** is at most 0.5;
- the **total scaled size** is at most 1.2;

Offline bins

Each bin (a set of items of size ≤ 1) contains **bonus items**:

- either no item of size $> 1/2$ and at most 5 items with bonus at most 0.1 each (actually the total is < 0.3),
- or one item of size $> 1/2$ and at most 2 items with bonus at most 0.1 total.

- Thus the **total bonus** is at most 0.5;
- the **total scaled size** is at most 1.2;
- the **total weight** is at most 1.7.

First Fit bins

- No item in a later bin fits into any previous bin.
- There is at most one bin of size $\leq 1/2$.
- There is at most one bin of size $< 2/3$ with at least two items.

First Fit bins

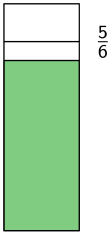
- No item in a later bin fits into any previous bin.
- There is at most one bin of size $\leq 1/2$.
- There is at most one bin of size $< 2/3$ with at least two items.
- Bins of size $\geq 5/6$ have weight ≥ 1.
- Bins with an item of size $> 1/2$ have weight ≥ 1.

First Fit bins

- No item in a later bin fits into any previous bin.
- There is at most one bin of size $\leq 1/2$.
- There is at most one bin of size $< 2/3$ with at least two items.
- Bins of size $\geq 5/6$ have weight ≥ 1.
- Bins with an item of size $> 1/2$ have weight ≥ 1.
- For the remaining bins with sizes in $(2/3, 5/6)$ we use **amortization**.
  We show that the scaled size of a bin plus the bonus of the **following** such bin is ≥ 1.

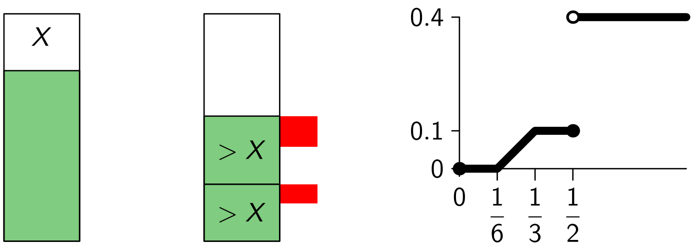
Amortization

For each bins with size in $(2/3, 5/6)$, at least two items, and no item $> 1/2$ we show that the scaled size of this bin plus the bonus of the **following** such bin is ≥ 1.



Amortization

For each bins with size in $(2/3, 5/6)$, at least two items, and no item $> 1/2$ we show that the scaled size of this bin plus the bonus of the **following** such bin is ≥ 1.



The result

$$FF(I) - 3 \leq w(I) \leq 1.7 \cdot OPT(I)$$

The result

$$FF(I) - 3 \leq w(I) \leq 1.7 \cdot OPT(I)$$

Theorem

First Fit has **asymptotic** approximation ratio 1.7.

$$FF(I) \leq 1.7 \cdot OPT(I) + 3$$

- First Fit is a special case of Best Fit
- The first item does not fit into any previous bin.
- If the first item is $\leq 1/2$ then also the second item does not fit into any previous bin.

- First Fit is a special case of Best Fit
- The first item does not fit into any previous bin.
- If the first item is $\leq 1/2$ then also the second item does not fit into any previous bin.
- We **close bins one by one**, paying 1 at each step.
  We always **close the largest bin**.
- The **bonus of one of the open bins** may be used to pay for the cost of closing **the previously closed bins**.

Amortization for Best Fit

The boss

One of the open bins is the **boss**.
It is always one of the two oldest bins.

Amortization for Best Fit

The boss

One of the open bins is the **boss**.
It is always one of the two oldest bins.

When a regular bin is closed

it has to pay the cost **itself**.

Amortization for Best Fit

The boss

One of the open bins is the **boss**.
It is always one of the two oldest bins.

When a regular bin is closed

it has to pay the cost **itself**.

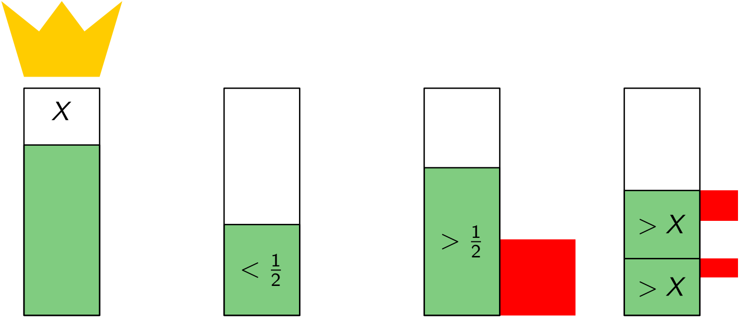
When the boss is closed

- We choose the new boss and
- we use also **its bonus** to pay the cost.

Choice of the new boss

The new boss is

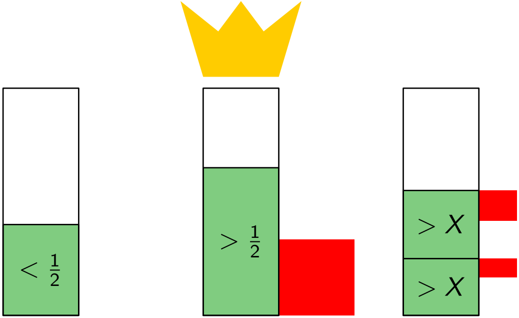
- the oldest bin, unless it has a single item and its size is $\leq 1/2$;
- the second oldest bin otherwise.



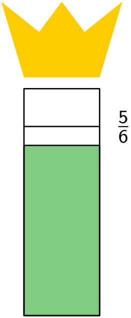
Choice of the new boss

The new boss is

- the oldest bin, unless it has a single item and its size is $\leq 1/2$;
- the second oldest bin otherwise.

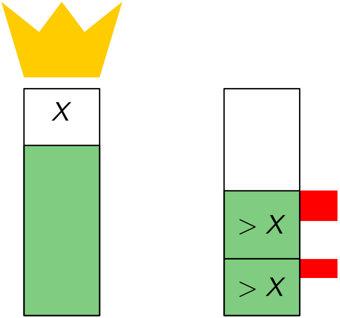


The boss is closed



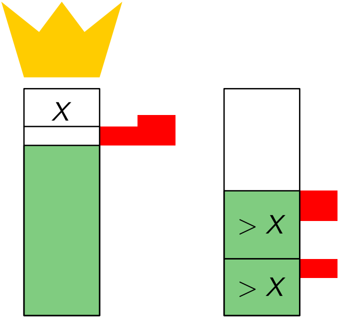
- The boss needs size $5/6$ to pay for himself. If it is smaller, then:

The boss is closed



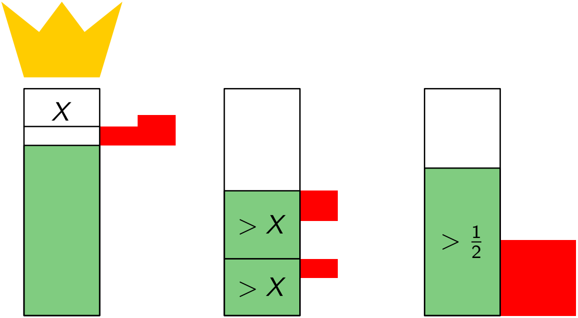
- The boss needs size $5/6$ to pay for himself. If it is smaller, then:
- Typically, the new boss has two items of size $> X$,

The boss is closed



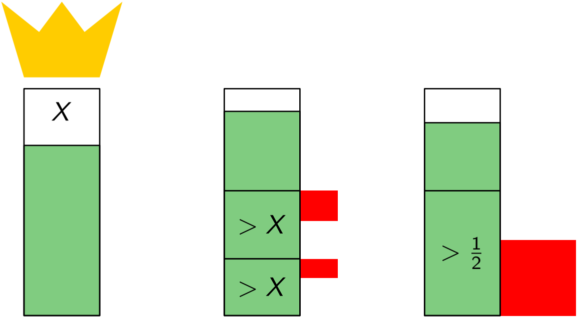
- The boss needs size $5/6$ to pay for himself. If it is smaller, then:
- Typically, the new boss has two items of size $> X$, which then have exactly the sufficient bonus.

The boss is closed



- The boss needs size $5/6$ to pay for himself. If it is smaller, then:
- Typically, the new boss has two items of size $> X$, which then have exactly the sufficient bonus.
- Or, even better, the new boss has an item of size $> \frac{1}{2}$.

A regular bin is closed



- If a regular bin has enough to pay for the boss, then it has enough for itself.

Results for Best Fit

At the end we have two bins left with total weight > 1.2. Thus

$$BF(I) \leq 1.7 \cdot OPT(I) + 0.7$$

Results for Best Fit

At the end we have two bins left with total weight > 1.2. Thus

$$BF(I) \leq 1.7 \cdot OPT(I) + 0.7$$

Bounded-space algorithms

- **At most** k **bins** are allowed to be **open**. A bin may be closed, i.e., later it cannot pack more items.

Results for Best Fit

At the end we have two bins left with total weight > 1.2. Thus

$$BF(I) \leq 1.7 \cdot OPT(I) + 0.7$$

Bounded-space algorithms

- **At most** k **bins** are allowed to be **open**. A bin may be closed, i.e., later it cannot pack more items.

k-bounded-space Best Fit

- The most full bin is closed when k bins are open and the next item does not fit into any of them.
- For $k \geq 2$, k-**bounded-space Best Fit** has **asymptotic** approximation ratio 1.7.

Modified weights

Classify the First Fit bins

- Big bins: **Size** $\geq 5/6$. No bonus, weight ≥ 1.

Modified weights

Classify the First Fit bins

- Big bins: **Size** $\geq 5/6$. No bonus, weight ≥ 1.
- Dedicated bins: **A single item**. Bonus 0.4, weight > 1.

Modified weights

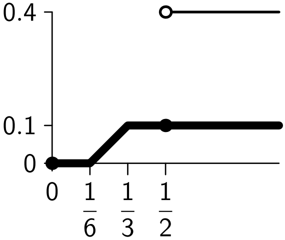
Classify the First Fit bins

- Big bins: **Size** $\geq 5/6$. No bonus, weight ≥ 1.
- Dedicated bins: **A single item**. Bonus 0.4, weight > 1. (Actually, 1 bin may have an item $\leq 1/2$, needs to have a smaller bonus. Still average weight > 1.)

Modified weights

Classify the First Fit bins

- Big bins: **Size** $\geq 5/6$. No bonus, weight ≥ 1.
- Dedicated bins: **A single item**. Bonus 0.4, weight > 1. (Actually, 1 bin may have an item $\leq 1/2$, needs to have a smaller bonus. Still average weight > 1.)
- **Common bins:** The rest. Bonus as before, but only 0.1 for items $> 1/2$. We need to prove that these C bins have **total weight** $\geq C - 0.2$.



Common bins

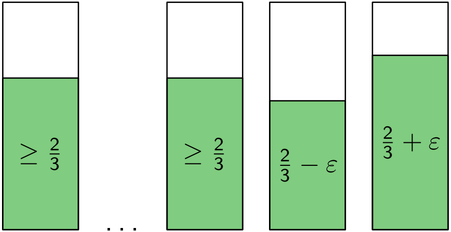
Distinguish a few cases:

- No bin of size $< 2/3$: Amortization as before.

Common bins

Distinguish a few cases:

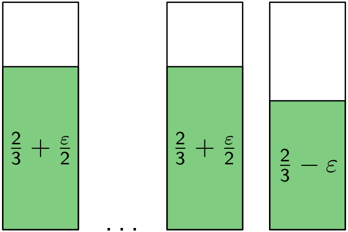
- No bin of size $< 2/3$: Amortization as before.
- A bin of size $< 2/3$, but it is not the last common bin: Easy fix.



Common bins

Distinguish a few cases:

- No bin of size $< 2/3$: Amortization as before.
- A bin of size $< 2/3$, but it is not the last common bin: Easy fix.
- The last common bin has size $< 2/3$: Harder, but works for large OPT. Some cases solved separately.



Last 0.1

Now we have $FF(I) - 0.2 < w(I) \leq 1.7 \cdot OPT(I)$
thus $FF(I) \leq 1.7 \cdot OPT(I) + 0.1$.

Last 0.1

Now we have $FF(I) - 0.2 < w(I) \leq 1.7 \cdot OPT(I)$
thus $FF(I) \leq 1.7 \cdot OPT(I) + 0.1$.

It remains to handle the case $OPT(I) \equiv 7 \pmod{10}$.

Last 0.1

Now we have $FF(I) - 0.2 < w(I) \leq 1.7 \cdot OPT(I)$
thus $FF(I) \leq 1.7 \cdot OPT(I) + 0.1$.

It remains to handle the case $OPT(I) \equiv 7 \pmod{10}$.

- Each OPT bin contain an item from a dedicated bin.

Last 0.1

Now we have $FF(I) - 0.2 < w(I) \leq 1.7 \cdot OPT(I)$
thus $FF(I) \leq 1.7 \cdot OPT(I) + 0.1$.

It remains to handle the case $OPT(I) \equiv 7 \pmod{10}$.

- Each OPT bin contain an item from a dedicated bin.
- Each OPT bin can contain **at most one item from a common bin with two items**.
- Parity argument: Some OPT bin contains **no such item**.

Last 0.1

Now we have $FF(I) - 0.2 < w(I) \leq 1.7 \cdot OPT(I)$
thus $FF(I) \leq 1.7 \cdot OPT(I) + 0.1$.

It remains to handle the case $OPT(I) \equiv 7 \pmod{10}$.

- Each OPT bin contain an item from a dedicated bin.
- Each OPT bin can contain **at most one item from a common bin with two items**.
- Parity argument: Some OPT bin contains **no such item**.
- **Remove the bonus of well-chosen two items** in such an OPT bin.

Last 0.1

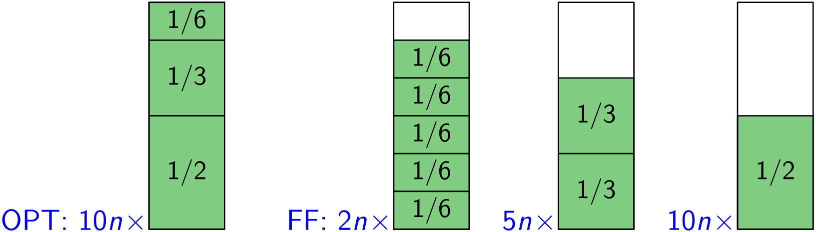
Now we have $FF(I) - 0.2 < w(I) \leq 1.7 \cdot OPT(I)$
thus $FF(I) \leq 1.7 \cdot OPT(I) + 0.1$.

It remains to handle the case $OPT(I) \equiv 7 \pmod{10}$.

- Each OPT bin contain an item from a dedicated bin.
- Each OPT bin can contain **at most one item from a common bin with two items**.
- Parity argument: Some OPT bin contains **no such item**.
- **Remove the bonus of well-chosen two items** in such an OPT bin. Then
  - the weight of this bin is at most 1.6 and thus $w(I) \leq 1.7 \cdot OPT(I) - 0.1$;
  - the analysis for the common bins still holds, since the items with removed bonus are in a bin with two more items.

Idealized example revisited

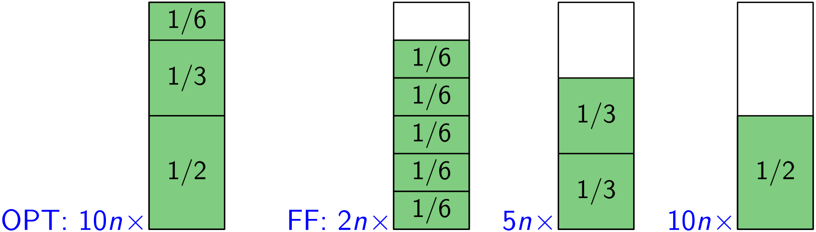
Assume that the algorithm cannot have bins of size exactly 1.



This can be achieved by changing the item sizes by a small amount and allowing OPT one extra bin.

Idealized example revisited

Assume that the algorithm cannot have bins of size exactly 1.



This can be achieved by changing the item sizes by a small amount and allowing OPT one extra bin.
We use this as a black box to get tight bounds.

Lower bounds

Suppose that $OPT = 10k + i$, $i = 0, \ldots, 9$.
Then the lower and upper bounds for First Fit are:

$i =$	0	1	2	3	4	5	6	7	8	9
$FF \geq 17k +$	-1	1	3	4	6	8	10	11	13	15
$FF \leq \lfloor 17k + 1.7i \rfloor = 17k +$	0	1	3	5	6	8	10	11	13	15

Lower bounds

Suppose that $OPT = 10k + i$, $i = 0, \ldots, 9$.
Then the lower and upper bounds for First Fit are:

$i =$	0	1	2	3	4	5	6	7	8	9
$FF \geq 17k +$	-1	1	3	4	6	8	10	11	13	15
$FF \leq \lfloor 17k + 1.7i \rfloor$ $= 17k +$	0	1	3	5	6	8	10	11	13	15

Conclusions

We have shown

$$\mathrm{FF} \leq 1.7 \cdot OPT$$

Improves previous $\mathrm{FF} \leq 1.7 \cdot OPT + 0.7$ and $\mathrm{FF} \leq 1.7143 \cdot OPT$.

Conclusions

We have shown

$$\mathrm{FF} \leq 1.7 \cdot OPT$$

Improves previous $\mathrm{FF} \leq 1.7 \cdot OPT + 0.7$ and $\mathrm{FF} \leq 1.7143 \cdot OPT$.

Open problems

- The **absolute** approximation ratio of **Best Fit**.

Conclusions

We have shown

$$\text{FF} \leq 1.7 \cdot OPT$$

Improves previous $\text{FF} \leq 1.7 \cdot OPT + 0.7$ and $\text{FF} \leq 1.7143 \cdot OPT$.

Open problems

- The **absolute** approximation ratio of **Best Fit**.
- The **absolute** approximation ratio of **bounded space Best Fit**.

Conclusions

We have shown

$$\text{FF} \leq 1.7 \cdot OPT$$

Improves previous $\text{FF} \leq 1.7 \cdot OPT + 0.7$ and $\text{FF} \leq 1.7143 \cdot OPT$.

Open problems

- The **absolute** approximation ratio of **Best Fit**.
- The **absolute** approximation ratio of **bounded space Best Fit**.
- The **gap for First Fit** for $OPT \equiv 0, 3 \pmod{10}$.

Conclusions

We have shown

$$\text{FF} \leq 1.7 \cdot OPT$$

Improves previous $\text{FF} \leq 1.7 \cdot OPT + 0.7$ and $\text{FF} \leq 1.7143 \cdot OPT$.

Open problems

- The **absolute** approximation ratio of **Best Fit**.
- The **absolute** approximation ratio of **bounded space Best Fit**.
- The **gap for First Fit** for $OPT \equiv 0, 3 \pmod{10}$.
- General online algorithms.
  The best bounds are 1.54037 and 1.58889.

Conclusions

We have shown

$$\mathrm{FF} \leq 1.7 \cdot OPT$$

Improves previous $\mathrm{FF} \leq 1.7 \cdot OPT + 0.7$ and $\mathrm{FF} \leq 1.7143 \cdot OPT$.

Open problems

- The **absolute** approximation ratio of **Best Fit**.
- The **absolute** approximation ratio of **bounded space Best Fit**.
- The **gap for First Fit** for $OPT \equiv 0, 3 \pmod{10}$.
- General online algorithms.
  The best bounds are 1.54037 and 1.58889.

THANK YOU!