# Online Colored Bin Packing

Martin Böhm, Jiří Sgall, **Pavel Veselý**

Computer Science Institute of Charles University, Prague, Czech Republic.

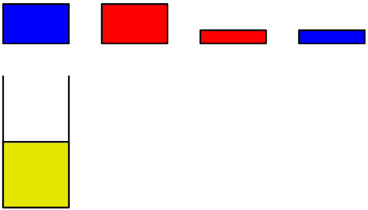
Trends in Online Algorithms 2014, July 7

# Colored Bin Packing

- Bin Packing
  - Input: items of sizes in $[0, 1]$
  - Goal: pack items into the minimum number of unit capacity bins

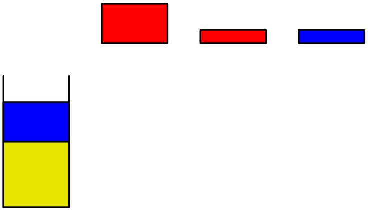
# Colored Bin Packing

- Bin Packing
  - Input: items of sizes in $[0, 1]$
  - Goal: pack items into the minimum number of unit capacity bins
- Colored Bin Packing
  - Each item has a color
  - Two items of the same color cannot be one on the other
  - Defined by [Balogh et al. '12] for two colors as Black and White Bin Packing



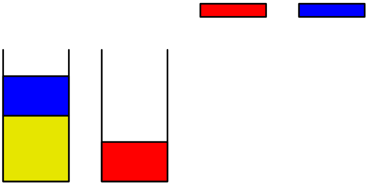
# Colored Bin Packing

- Bin Packing
  - Input: items of sizes in $[0, 1]$
  - Goal: pack items into the minimum number of unit capacity bins
- Colored Bin Packing
  - Each item has a color
  - Two items of the same color cannot be one on the other
  - Defined by [Balogh et al. '12] for two colors as Black and White Bin Packing



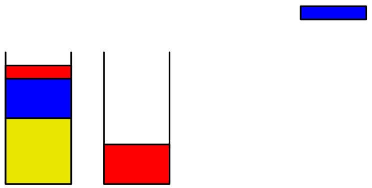
# Colored Bin Packing

- Bin Packing
  - Input: items of sizes in $[0,1]$
  - Goal: pack items into the minimum number of unit capacity bins
- Colored Bin Packing
  - Each item has a color
  - Two items of the same color cannot be one on the other
  - Defined by [Balogh et al. '12] for two colors as Black and White Bin Packing



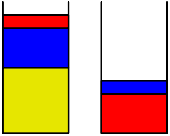
# Colored Bin Packing

- Bin Packing
  - Input: items of sizes in $[0, 1]$
  - Goal: pack items into the minimum number of unit capacity bins
- Colored Bin Packing
  - Each item has a color
  - Two items of the same color cannot be one on the other
  - Defined by [Balogh et al. '12] for two colors as Black and White Bin Packing



# Colored Bin Packing

- Bin Packing
  - Input: items of sizes in $[0, 1]$
  - Goal: pack items into the minimum number of unit capacity bins
- Colored Bin Packing
  - Each item has a color
  - Two items of the same color cannot be one on the other
  - Defined by [Balogh et al. '12] for two colors as Black and White Bin Packing

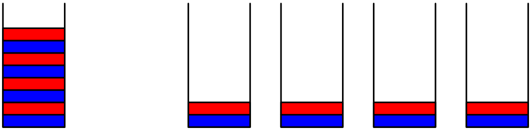


# Offline vs. restricted offline settings

- Offline
  - Items are given in advance
  - We can pack in any order

# Offline vs. restricted offline settings

- Offline
  - Items are given in advance
  - We can pack in any order
- Restricted offline
  - Items are given as a sequence
  - We have to pack them in the given order
  - Optimum can differ from the unrestricted offline case:
    - $n$ blue and then $n$ red, all of size zero



# Competitive ratio of an online algorithm

- For an input list of items $L$:
  - $ALG(L) = \#$ of bins used by $ALG$
  - $OPT(L) =$ restricted offline optimum
- $ALG$ is *absolutely* $r$-competitive if:
  - for any $L$ it holds $ALG(L) \leq r \cdot OPT(L)$

# Competitive ratio of an online algorithm

- For an input list of items $L$:
  - $ALG(L) = \#$ of bins used by $ALG$
  - $OPT(L) =$ restricted offline optimum
- $ALG$ is *absolutely* $r$-competitive if:
  - for any $L$ it holds $ALG(L) \leq r \cdot OPT(L)$
- $ALG$ is *asymptotically* $r$-competitive if:
  - $ALG(L) \leq r \cdot OPT(L) + o(OPT(L))$

# Competitive ratio of an online algorithm

- For an input list of items $L$:
  - $ALG(L) = \#$ of bins used by $ALG$
  - $OPT(L) =$ restricted offline optimum
- $ALG$ is *absolutely* $r$-competitive if:
  - for any $L$ it holds $ALG(L) \leq r \cdot OPT(L)$
- $ALG$ is *asymptotically* $r$-competitive if:
  - $ALG(L) \leq r \cdot OPT(L) + o(OPT(L))$
- $ALG$ has the *competitive ratio* $r$ if
  - it is $r$-competitive
  - it is not $r'$-competitive for $r' < r$

# Notation

- Level of a bin = cumulative size of all items in the bin
- $c$-item = an item of color $c$
- $c$-bin = a bin with a $c$-item on the top
- Example: red bin: 

# Lower bound on the restricted offline optimum

- Sum of items sizes $LB_1$
- Maximal color discrepancy $LB_2$
  - 10 white, 2 red and 10 white must be packed into $\geq 18$ bins

# Lower bound on the restricted offline optimum

- Sum of items sizes $LB_1$
- Maximal color discrepancy $LB_2$
  - 10 white, 2 red and 10 white must be packed into $\geq 18$ bins
  - Discrepancy for a color $c$ on an interval of the input sequence:

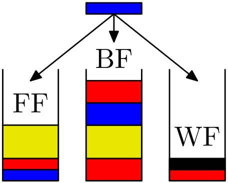
$$\#\text{ of } c\text{-items} - \#\text{ of items of other colors}$$

# Any Fit algorithms

- Opens a bin if it is really necessary

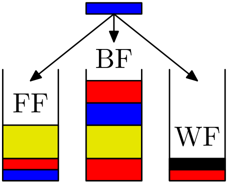
# Any Fit algorithms

- Opens a bin if it is really necessary
- Main variants:
  - First Fit (FF): chooses the *first* bin in which an incoming item fits
  - Best Fit (BF): chooses the bin with the *highest* level
  - Worst Fit (WF): chooses the bin with the *lowest* level



# Any Fit algorithms

- Opens a bin if it is really necessary
- Main variants:
  - First Fit (FF): chooses the *first* bin in which an incoming item fits
  - Best Fit (BF): chooses the bin with the *highest* level
  - Worst Fit (WF): chooses the bin with the *lowest* level




- We study both general and parametric cases
  - *Parametric case*: for a real $d \geq 2$ the items have size at most $\frac{1}{d}$

# Black and White Bin Packing

- [Balogh et al. '12 and '13], [Dósa and Epstein '14]
  - Lower bound of 2 on competitiveness of all online algorithms

# Black and White Bin Packing

- [Balogh et al. '12 and '13], [Dósa and Epstein '14]
  - Lower bound of 2 on competitiveness of all online algorithms
  - Competitiveness of algorithms – previous results:

| Algorithm | Lower bound | Upper bound |
|---|---:|---:|
| First Fit | 3 | 5 |
| Best Fit | 3 | 5 |
| Worst Fit [parametric case] | 3 $[1 + \frac{d}{d-1}]$ | 5 |
| Pseudo [parametric case] | 3 $[1 + \frac{d}{d-1}]$ | 3 $[1 + \frac{d}{d-1}]$ |

# Black and White Bin Packing

- [Balogh et al. '12 and '13], [Dósa and Epstein '14]
  - Lower bound of 2 on competitiveness of all online algorithms
  - Competitiveness of algorithms – previous and our results:

| Algorithm | Lower bound | Upper bound | |
|---|---|---|---|
| First Fit | 3 | 3 | 5 |
| Best Fit | 3 | 3 | 5 |
| Worst Fit [parametric case] | 3 $[1+\frac{d}{d-1}]$ | 3 $[1+\frac{d}{d-1}]$ | 5 |
| Pseudo [parametric case] | 3 $[1+\frac{d}{d-1}]$ | 3 $[1+\frac{d}{d-1}]$ | |

- Our results
  - Any Fit algorithms are absolutely 3-competitive
  - Worst Fit for items of size $\leq \frac{1}{d}$ has ratio exactly $1+\frac{d}{d-1}$

# Colored Bin Packing

- [Dósa and Epstein '14] independently of us
  - Lower bound of 2 on competitiveness of all online algorithms
  - For zero-size items
    - *Asymptotic* lower bound 1.5
    - 2-competitive algorithm
  - 4-competitive algorithm for items of any size

# Colored Bin Packing

- [Dósa and Epstein '14] independently of us
  - Lower bound of 2 on competitiveness of all online algorithms
  - For zero-size items
    - *Asymptotic* lower bound 1.5
    - 2-competitive algorithm
  - 4-competitive algorithm for items of any size
- Our results
  - For zero-size items
    - Restricted offline optimum = maximal color discrepancy

# Colored Bin Packing

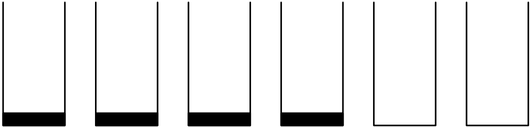
- [Dósa and Epstein '14] independently of us
  - Lower bound of 2 on competitiveness of all online algorithms
  - For zero-size items
    - *Asymptotic* lower bound 1.5
    - 2-competitive algorithm
  - 4-competitive algorithm for items of any size
- Our results
  - For zero-size items
    - Restricted offline optimum = maximal color discrepancy
    - Optimal 1.5-competitive algorithm – uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
    - Lower bound of $\lceil 1.5 \cdot OPT \rceil$ for all online algorithms
  - 3.5-competitive algorithm for items of any size
    - $(1.5 + \frac{d}{d-1})$-competitive in the parametric case

# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:

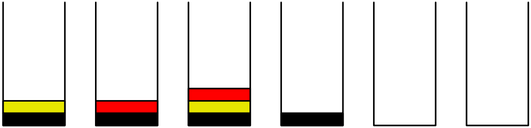
# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:



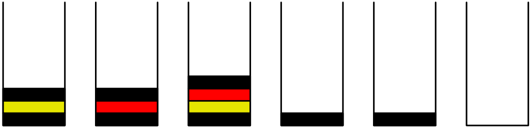
# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:



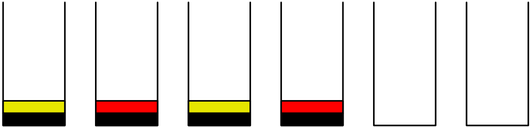
# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:



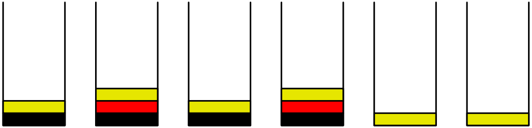
# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:



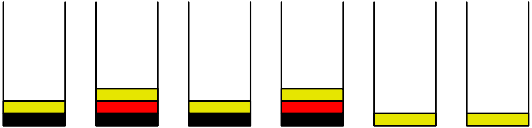
# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:



# Lower bound 1.5 for zero-size items

- Let $n$ be the optimum
- The adversary sends the instance in phases
- In each phase:
  - # of black bins increases,
  - or we get $\lceil 1.5 \cdot n \rceil$ bins
- Example for $n = 4$:



- A little bit more complicated for an odd $n$ to get $\lceil 1.5 \cdot n \rceil$ bins

# Optimal algorithm for zero-size items

- **Balancing Any Fit** (BAF)
- Uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
- $N_c = \#$ of $c$-bins
- Current discrepancy of a color $c$
  - $CD_c =$ max. discrepancy on an interval ending just before the incoming item

# Optimal algorithm for zero-size items

- **Balancing Any Fit** (BAF)
- Uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
- $N_c = \#$ of $c$-bins
- Current discrepancy of a color $c$
  - $CD_c =$ max. discrepancy on an interval ending just before the incoming item
- Main invariant for a color $c$
  - $N_c \leq CD_c + \lceil 0.5 \cdot OPT \rceil$

# Optimal algorithm for zero-size items

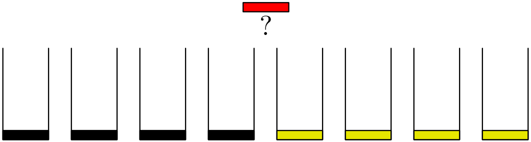
- **Balancing Any Fit** (BAF)
- Uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
- $N_c = \#$ of $c$-bins
- Current discrepancy of a color $c$
  - $CD_c =$ max. discrepancy on an interval ending just before the incoming item
- Main invariant for a color $c$
  - $N_c \leq CD_c + \lceil 0.5 \cdot OPT \rceil$
- BAF mostly puts an incoming $c$-item into a bin of the most frequent other color

# Optimal algorithm for zero-size items

- **Balancing Any Fit** (BAF)
- Uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
- $N_c = \#$ of $c$-bins
- Current discrepancy of a color $c$
  - $CD_c$ = max. discrepancy on an interval ending just before the incoming item
- Main invariant for a color $c$
  - $N_c \leq CD_c + \lceil 0.5 \cdot OPT \rceil$
- BAF mostly puts an incoming $c$-item into a bin of the most frequent other color
  - Exception: two colors have more than $\lceil 0.5 \cdot OPT \rceil$ bins

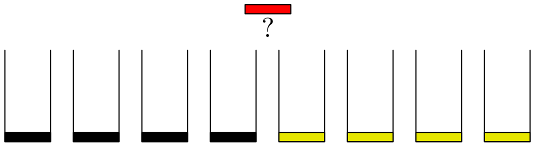
# BAF: two colors have more than $\lceil 0.5 \cdot OPT \rceil$ bins

- Let these colors be black and yellow
- Example with $OPT = 5$ and $\lceil 1.5 \cdot OPT \rceil = 8$
  - Suppose that $CD_{\mathrm{black}} = 1$ and $CD_{\mathrm{yellow}} = 1$
  - Thus $N_{\mathrm{b}} = CD_{\mathrm{b}} + \lceil 0.5 \cdot OPT \rceil$ and $N_{\mathrm{y}} = CD_{\mathrm{y}} + \lceil 0.5 \cdot OPT \rceil$



# BAF: two colors have more than $\lceil 0.5 \cdot OPT \rceil$ bins

- Let these colors be black and yellow
- Example with $OPT = 5$ and $\lceil 1.5 \cdot OPT \rceil = 8$
  - Suppose that $CD_{\text{black}} = 1$ and $CD_{\text{yellow}} = 1$
  - Thus $N_{\text{b}} = CD_{\text{b}} + \lceil 0.5 \cdot OPT \rceil$ and $N_{\text{y}} = CD_{\text{y}} + \lceil 0.5 \cdot OPT \rceil$




- We need to prove that
  - $N_{\text{b}} < CD_{\text{b}} + \lceil 0.5 \cdot OPT \rceil$,
  - or $N_{\text{y}} < CD_{\text{y}} + \lceil 0.5 \cdot OPT \rceil$

# Algorithm Pseudo

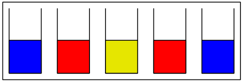
- First Fit, Best Fit and Worst Fit
  - Bad behavior for at least three colors

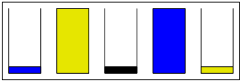
# Algorithm Pseudo

- First Fit, Best Fit and Worst Fit
  - Bad behavior for at least three colors
- Pseudo: 3.5-competitive algorithm for items of any size
  - Uses pseudo bins = bins of unlimited capacity
    - Divides them into unit capacity bins

# Algorithm Pseudo

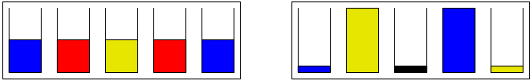
- First Fit, Best Fit and Worst Fit
  - Bad behavior for at least three colors
- Pseudo: 3.5-competitive algorithm for items of any size
  - Uses pseudo bins = bins of unlimited capacity
    - Divides them into unit capacity bins
  - Put an incoming item into a pseudo bin using BAF
  - Apply Next Fit in the pseudo bin





# Algorithm Pseudo

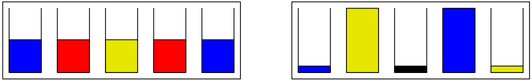
- Pseudo: 3.5-competitive algorithm for items of any size
  - Uses pseudo bins = bins of unlimited capacity
    - Divides them into unit capacity bins
  - Put an incoming item into a pseudo bin using BAF
  - Apply Next Fit in the pseudo bin



- Proof of 3.5-competitiveness
  - We pair all bins except one in each pseudo bin
    - Each pair has total volume of more than 1
    - # of paired bins is at most $2 \cdot OPT - 1$
  - # of non-paired bins $\leq$ # of pseudo bins
    - BAF uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
  - Altogether at most $3.5 \cdot OPT$ bins

# Algorithm Pseudo

- Pseudo: 3.5-competitive algorithm for items of any size
  - Uses pseudo bins = bins of unlimited capacity
    - Divides them into unit capacity bins
  - Put an incoming item into a pseudo bin using BAF
  - Apply Next Fit in the pseudo bin



- Proof of 3.5-competitiveness
  - We pair all bins except one in each pseudo bin
    - Each pair has total volume of more than 1
    - # of paired bins is at most $2 \cdot OPT - 1$
  - # of non-paired bins $\leq$ # of pseudo bins
    - BAF uses at most $\lceil 1.5 \cdot OPT \rceil$ bins
  - Altogether at most $3.5 \cdot OPT$ bins
- In the parametric case $(1.5 + \frac{d}{d-1})$-competitive

# Worst Fit in the parametric case for two colors

- Worst Fit is $(1 + \frac{d}{d-1})$-competitive
  - If all items have size $\leq \frac{1}{d}$ for a real $d \geq 2$
- Idea of the proof:
  - Big bins = bins with level $\geq \frac{d-1}{d}$
    - # of big bins is at most $\frac{d}{d-1} \cdot LB_1$
  - Small bins = bins with level $< \frac{d-1}{d}$
  - We bound # of small bins from above by $LB_2$

# Any Fit algorithms for two colors

- Any algorithm in the Any Fit family is absolutely 3-competitive
  - Similar proof, but more complicated
  - Big bins have level $\geq 0.5$ and small bins $< 0.5$
  - # of small bins cannot be bounded by color discrepancy $LB_2$

# Any Fit algorithms for two colors

- Any algorithm in the Any Fit family is absolutely 3-competitive
  - Similar proof, but more complicated
  - Big bins have level $\geq 0.5$ and small bins $< 0.5$
  - $\#$ of small bins cannot be bounded by color discrepancy $LB_2$
  - We assign bins into *chains*
    - Sequences of bins where the average level is $\geq 0.5$
  - We bound the number of bins not in chains by $LB_2$

# Conclusions

- For at least three colors
  - We have solved COLORED BIN PACKING for zero-size items
  - For items of any size we have 3.5-competitive algorithm
  - We have recently improved the lower bound to 2.5
- For two colors
  - We improved the upper bound on competitiveness of Any Fit algorithms
    - Tight for First Fit, Best Fit and Worst Fit

# Open problems

- Design a better than 3.5-competitive algorithm
- Or improve the lower bound of 2.5
- Prove that no Any Fit algorithm can be better than 3-competitive for two colors

# Open problems

- Design a better than 3.5-competitive algorithm
- Or improve the lower bound of 2.5
- Prove that no Any Fit algorithm can be better than 3-competitive for two colors
    - Or find a better one

Thank you for your attention