Online Bin Packing with Predictions

Spyros Angelopoulos

Work with Shahin Kamali (York) and Kimia Shadkami (Manitoba)

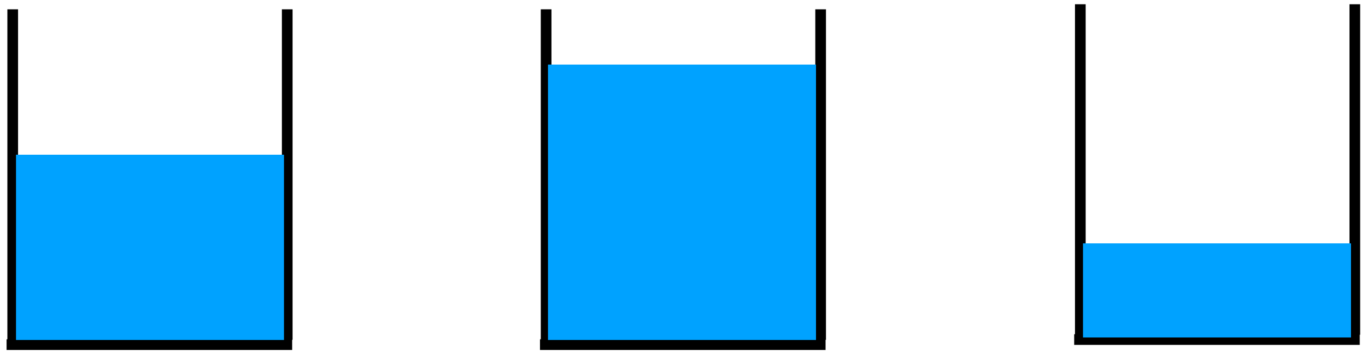
JFRO, Paris, February 2, 2024

Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity

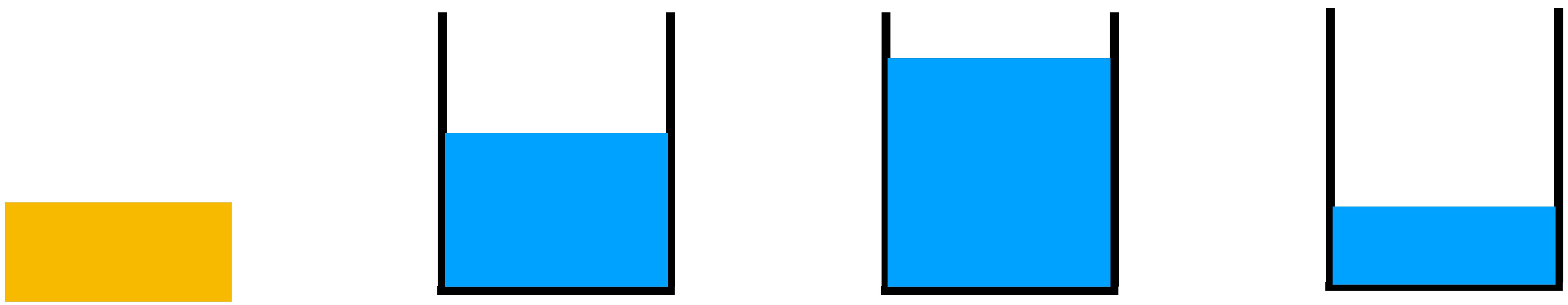
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



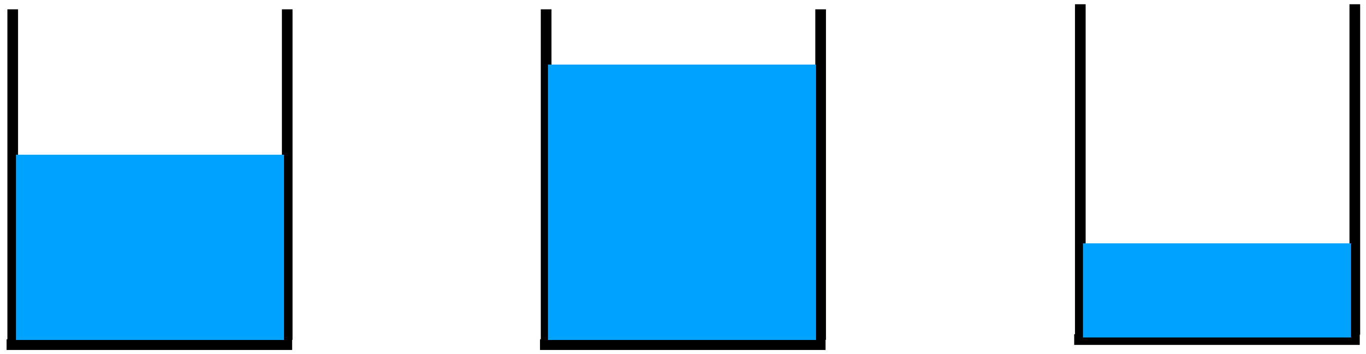
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



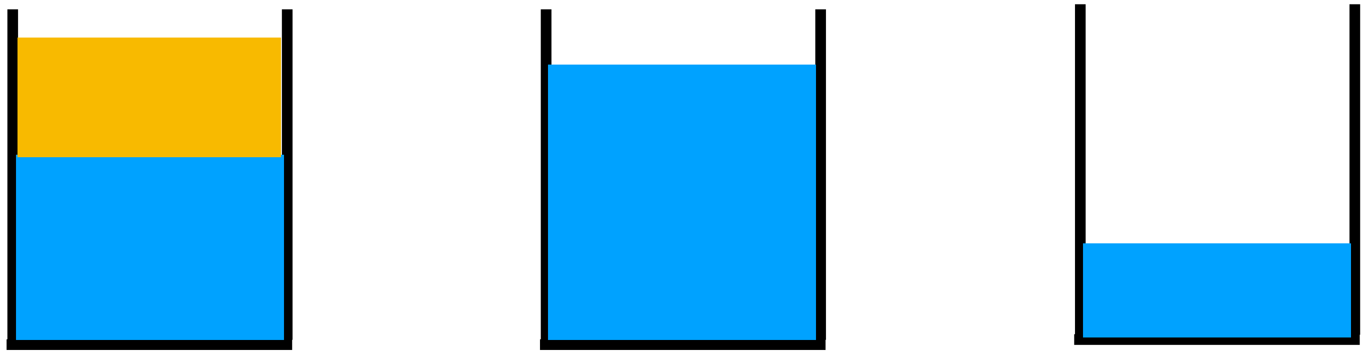
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



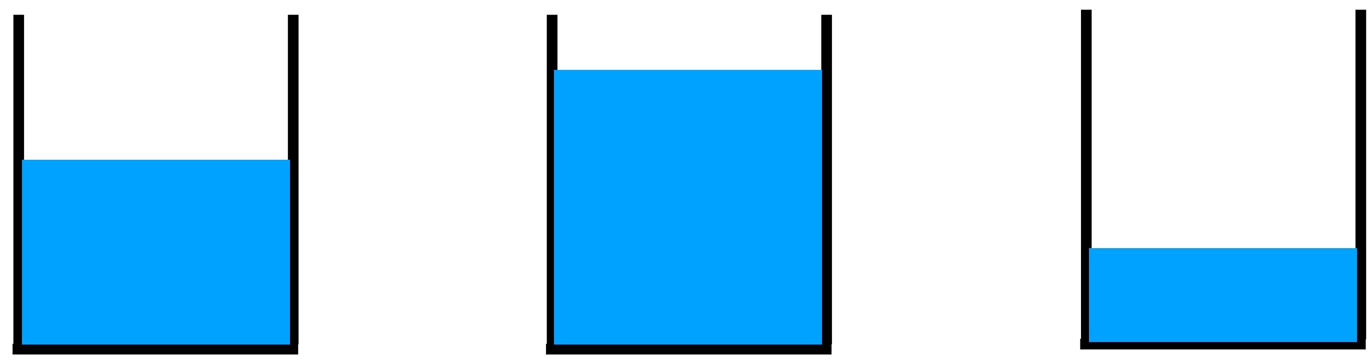
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



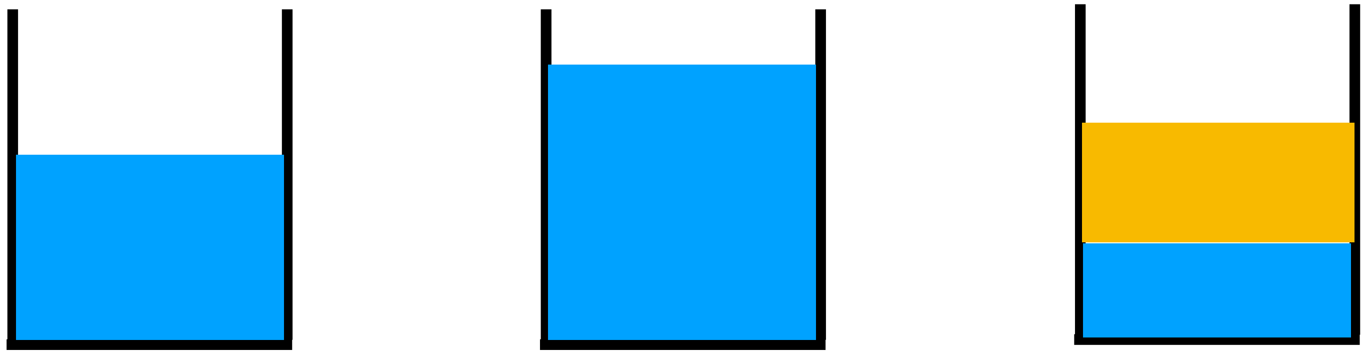
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



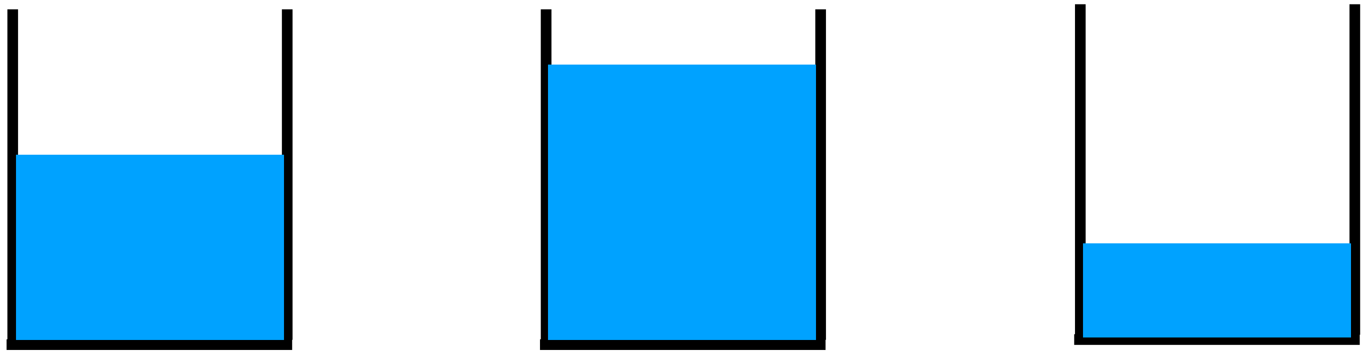
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



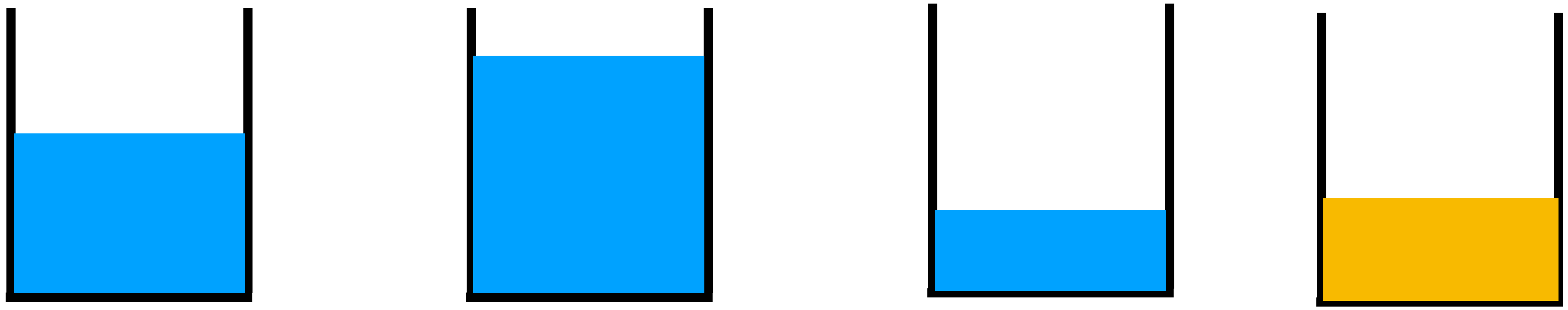
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



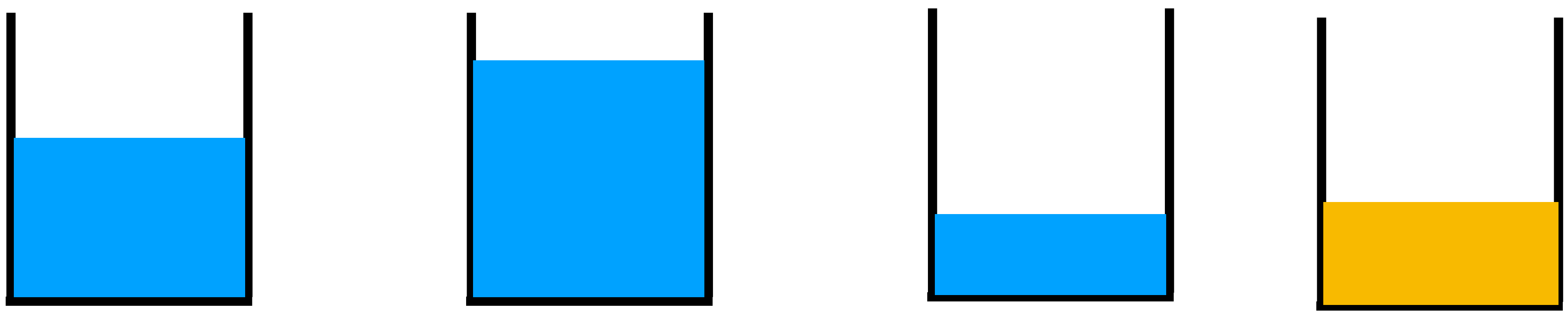
Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



Bin packing

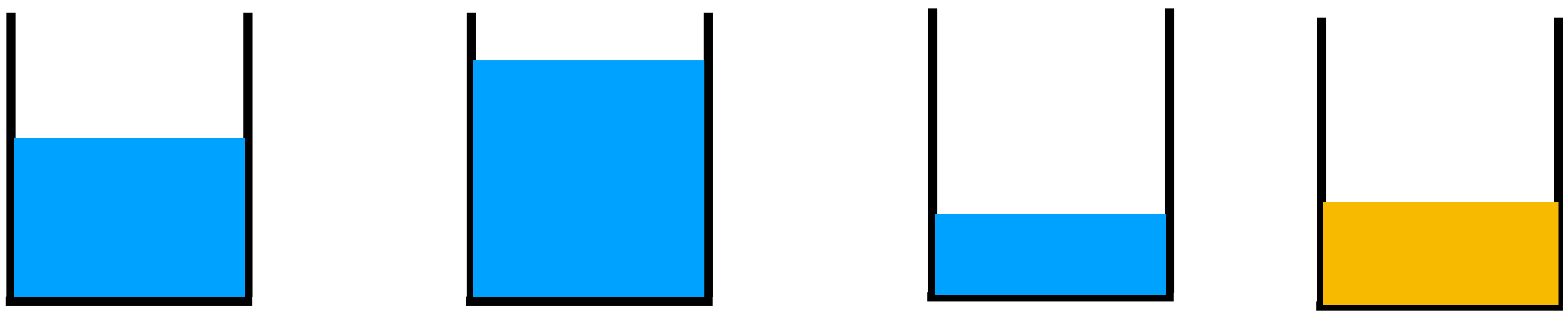
Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



Objective: Minimize the (asymptotic) competitive ratio

Bin packing

Setting: Pack a sequence of items (each with its own weight) into the minimum number of bins of a given capacity



Objective: Minimize the (asymptotic) competitive ratio

Many applications (e.g., cloud computing)

Some important results

Best known **upper** bound : 1.57829 [Balogh, Békési, Dósa, Epstein, Levin 2018]

Best known **lower** bound : 1.54278 [Balogh, Békési, Dósa, Epstein, Levin 2021]

FIRST-FIT, BEST-FIT have competitive ratio 1.7 [Johnson et al. 1974]

Some important results

Best known **upper** bound : 1.57829 [Balogh, Békési, Dósa, Epstein, Levin 2018]

Best known **lower** bound : 1.54278 [Balogh, Békési, Dósa, Epstein, Levin 2021]

FIRST-FIT, BEST-FIT have competitive ratio 1.7 [Johnson et al. 1974]

In practice, FIRST-FIT and BEST-FIT perform very well [Kamali and López-Ortiz 2015]

In practice, many competitively efficient algorithms do not perform as well as FIRST-FIT

Some important results

Best known **upper** bound : 1.57829 [Balogh, Békési, Dósa, Epstein, Levin 2018]

Best known **lower** bound : 1.54278 [Balogh, Békési, Dósa, Epstein, Levin 2021]

FIRST-FIT, BEST-FIT have competitive ratio 1.7 [Johnson et al. 1974]

In practice, FIRST-FIT and BEST-FIT perform very well [Kamali and López-Ortiz 2015]

In practice, many competitively efficient algorithms do not perform as well as FIRST-FIT

Enhance the standard model of so as to leverage some additional information about the input

Online algorithms with predictions

[Lykouris and Vassilvitskii 2018]

[Purohit et al. 2018]

Online algorithms with predictions

[Lykouris and Vassilvitskii 2018]

[Purohit et al. 2018]

Access to a **prediction** associated with the input which is inherently **erroneous**

The prediction has error η (unknown to the algorithm)

Online algorithms with predictions

[Lykouris and Vassilvitskii 2018]

[Purohit et al. 2018]

Access to a **prediction** associated with the input which is inherently **erroneous**

The prediction has error η (unknown to the algorithm)

Online algorithms with predictions

[Lykouris and Vassilvitskii 2018]

[Purohit et al. 2018]

Access to a **prediction** associated with the input which is inherently **erroneous**

The prediction has error η (unknown to the algorithm)

Robustness : competitive ratio
with *adversarial* error

Consistency : competitive ratio
with *no* error

Online algorithms with predictions

[Lykouris and Vassilvitskii 2018]

[Purohit et al. 2018]

Access to a **prediction** associated with the input which is inherently **erroneous**

The prediction has error η (unknown to the algorithm)

Online algorithms with predictions

[Lykouris and Vassilvitskii 2018]

[Purohit et al. 2018]

Access to a **prediction** associated with the input which is inherently **erroneous**

The prediction has error η (unknown to the algorithm)

Robustness : competitive ratio with *adversarial* error

Consistency : competitive ratio with *no* error

competitive ratio with error η

Predictions that are "learnable" (e.g., sampling of the input)

Algorithms that degrade "gently" with error

Theoretical and experimental results

Side note: advice complexity of bin packing

Competitive ratio of algorithms with access to bits of “additional information”

Side note: advice complexity of bin packing

Competitive ratio of algorithms with access to bits of “additional information”

- **Trusted advice :** [Boyar, Kamali, Larsen and López-Ortiz 2016]

  [Mikkelsen 2016]

  [A., Dürr, Kamali, Renault and Rosén 2018]

  Trade-offs between advice size and competitive ratio, for **error-free** advice

Side note: advice complexity of bin packing

Competitive ratio of algorithms with access to bits of “additional information”

- **Trusted advice :** [Boyar, Kamali, Larsen and López-Ortiz 2016]

  [Mikkelsen 2016]

  [A., Dürr, Kamali, Renault and Rosén 2018]

  Trade-offs between advice size and competitive ratio, for **error-free** advice

- **Untrusted advice:** [A., Dürr, Jin, Kamali, and Renault 2020]

  Consistency-robustness tradeoffs for advice of a given size

Bin packing with frequency predictions

We assume a *discrete* model: The bin capacity is a constant k, and each item has integral size in $[1,k]$

Bin packing with frequency predictions

We assume a *discrete* model: The bin capacity is a constant k, and each item has integral size in [1,k]

Prediction: *Frequencies* at which the items are requested in the sequence

Formally: for each size $x \in [1,k]$, the *frequency* $f_{x,\sigma}$ of x in the sequence σ is the number of items of size x in σ divided by the size of σ

Prediction error η: L_1 distance between the actual and the predicted frequencies

Bin packing with frequency predictions

We assume a *discrete* model: The bin capacity is a constant k, and each item has integral size in $[1,k]$

Prediction: *Frequencies* at which the items are requested in the sequence

Formally: for each size $x \in [1,k]$, the *frequency* $f_{x,\sigma}$ of x in the sequence σ is the number of items of size x in σ divided by the size of σ

Prediction error η: L_1 distance between the actual and the predicted frequencies

Frequency predictions are PAC-learnable

Profile of a bin packing sequence

Profile of a bin packing sequence

Fix a (large) constant M. We call the multiset that consists of $\lceil f_{x,\sigma} \cdot M \rceil$ items of size x

the **profile** of σ

We can compute the optimal packing of this profile set in $O(1)$ time

[Fukunaga and Korf 2007]

Profile of a bin packing sequence

Fix a (large) constant M. We call the multiset that consists of $\lceil f_{x,\sigma} \cdot M \rceil$ items of size x the **profile** of σ

We can compute the optimal packing of this profile set in $O(1)$ time [Fukunaga and Korf 2007]

Example: M=14, $k = 3, f_1 = 0.7, f_2 = 0.2, f_3 = 0.1$

Profile consists of 10 items of size 1, 3 items of size 2 and 2 items of size 3

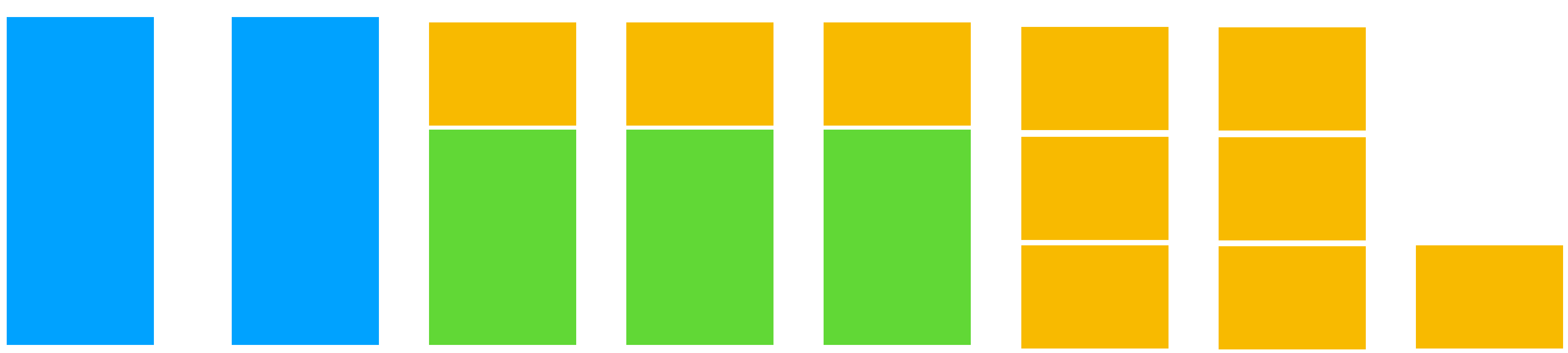
Profile of a bin packing sequence

Fix a (large) constant M. We call the multiset that consists of $\lceil f_{x,\sigma} \cdot M \rceil$ items of size x the **profile** of σ

We can compute the optimal packing of this profile set in $O(1)$ time [Fukunaga and Korf 2007]

Example: M=14, $k = 3, f_1 = 0.7, f_2 = 0.2, f_3 = 0.1$

Profile consists of 10 items of size 1, 3 items of size 2 and 2 items of size 3



Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of single bins (virtually)

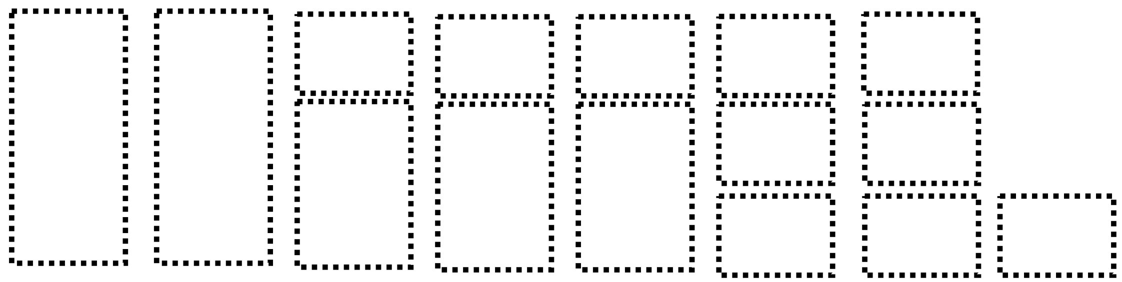
Items that were not predicted to appear are packed separately using FIRST-FIT

Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of single bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

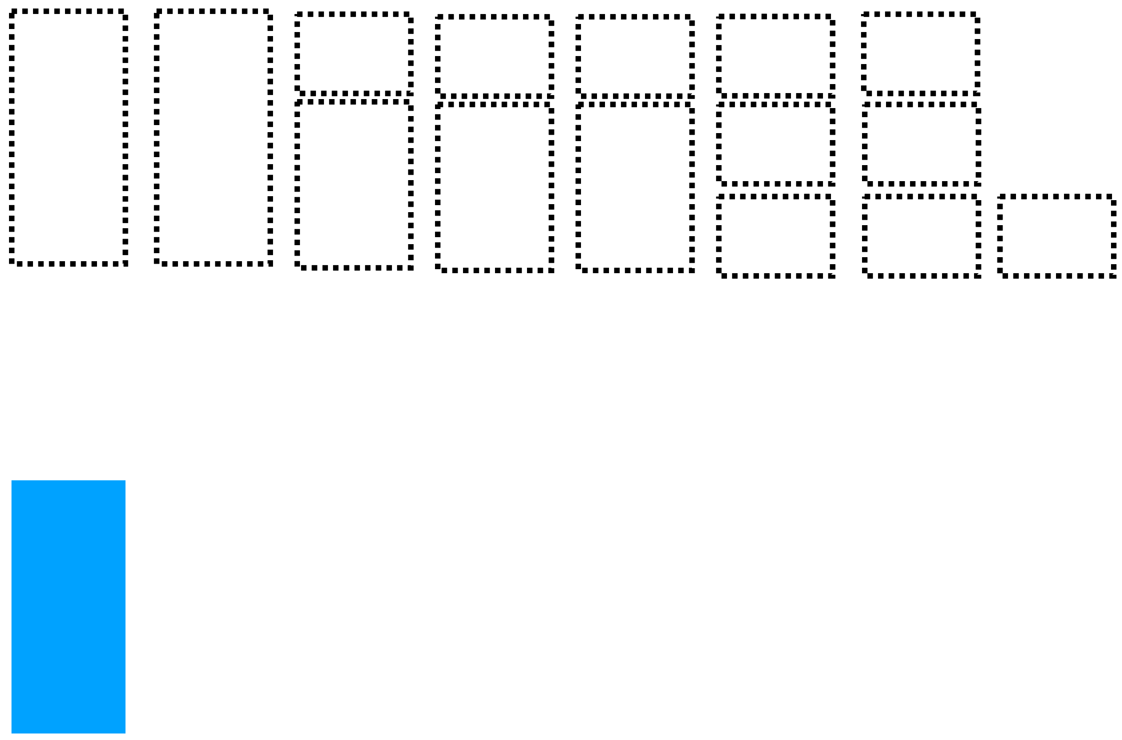


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

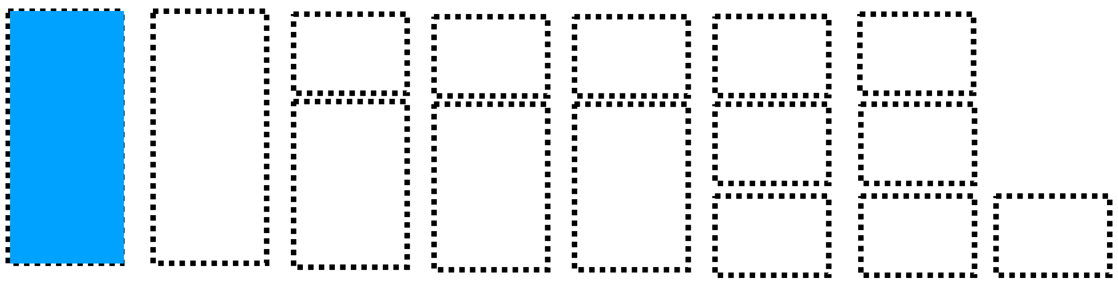


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

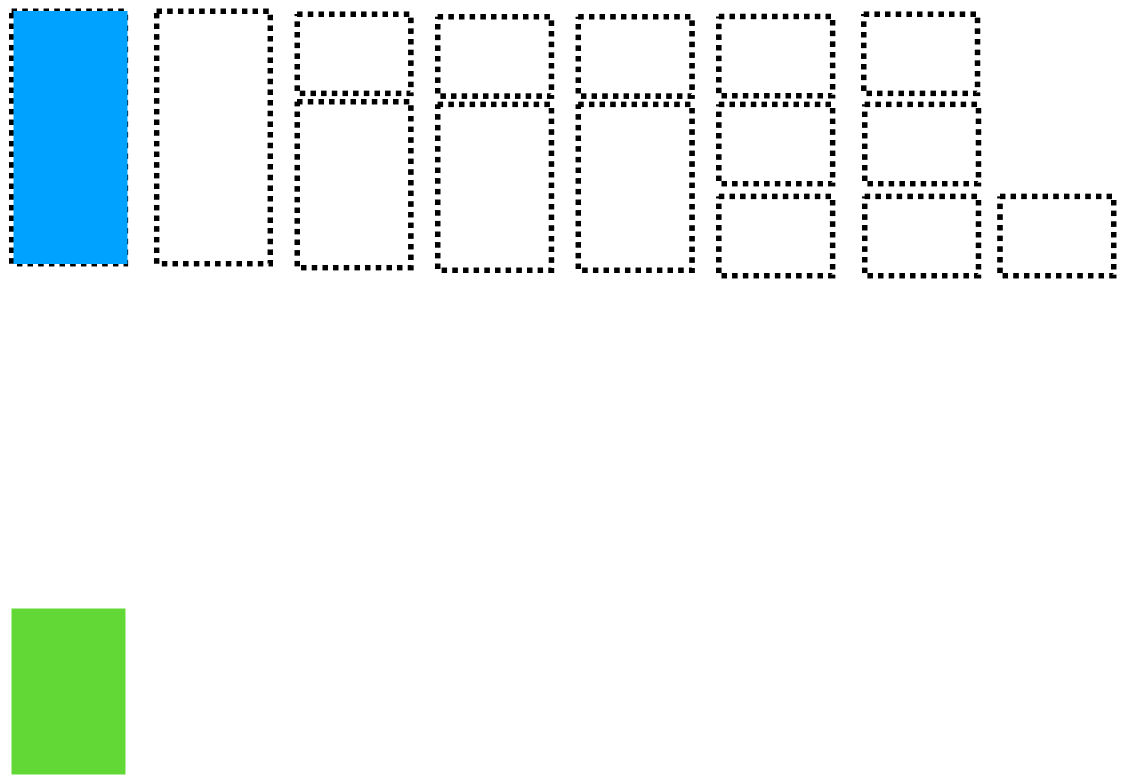


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

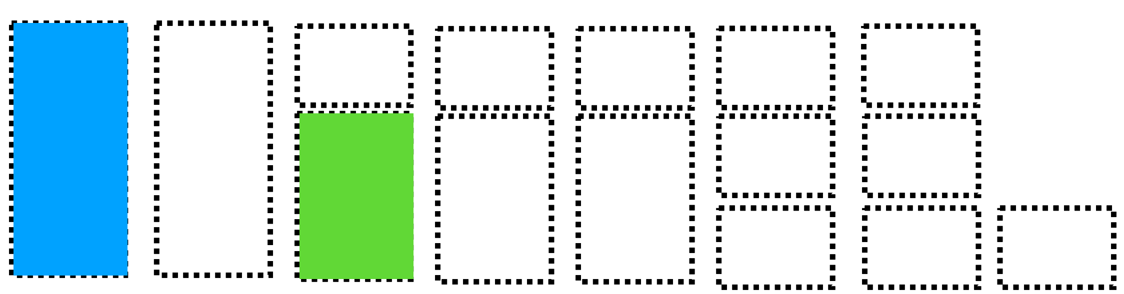


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

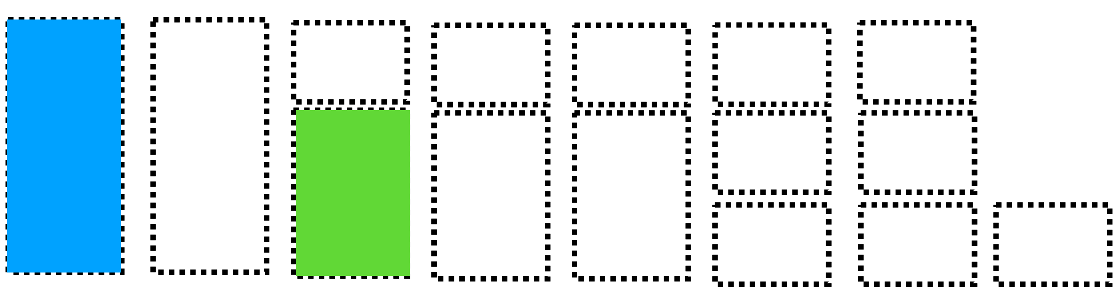


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

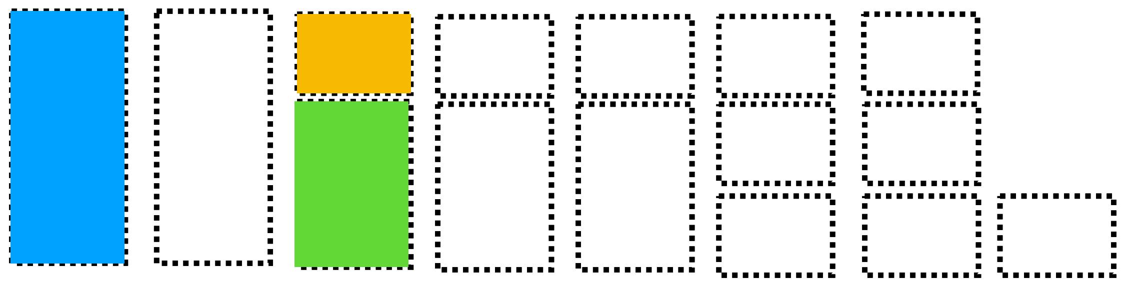


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

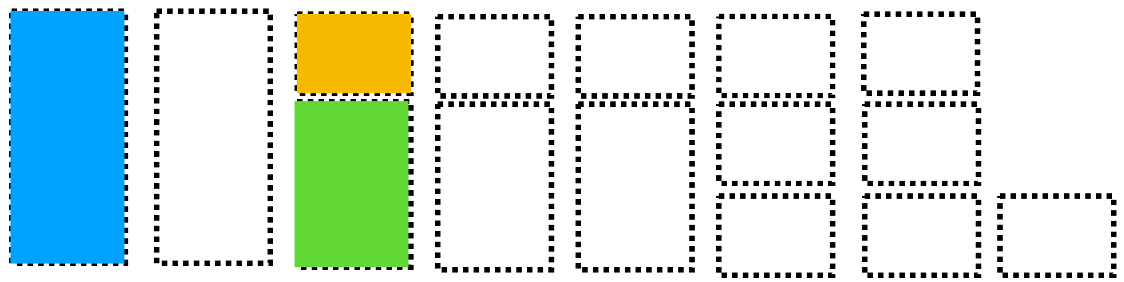


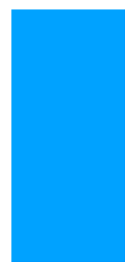
Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT



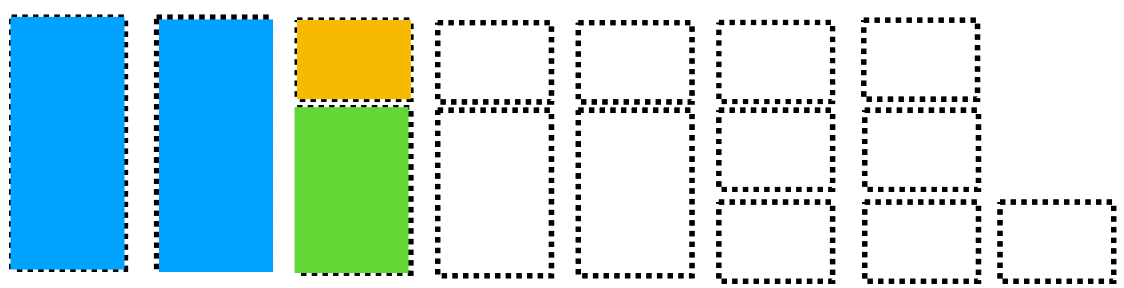


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

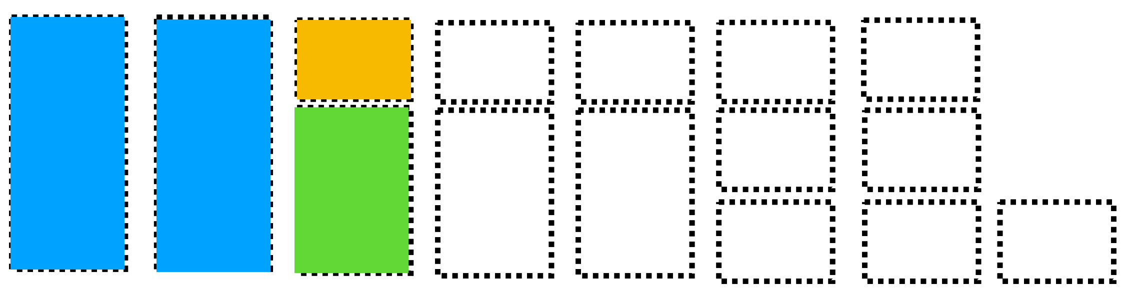


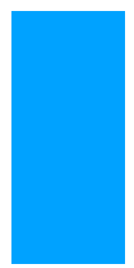
Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT



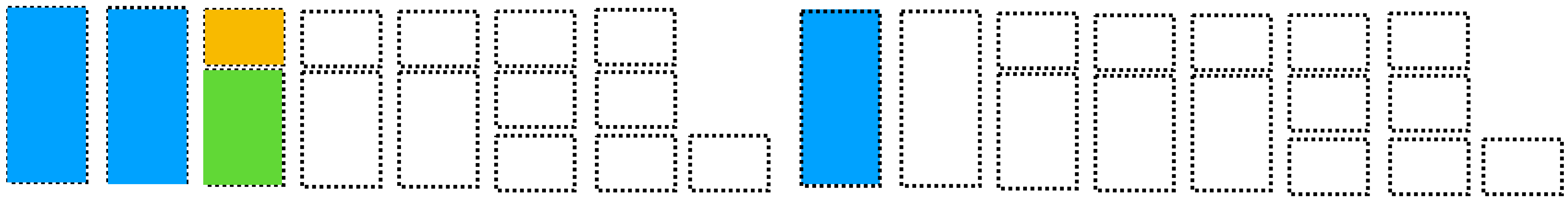


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

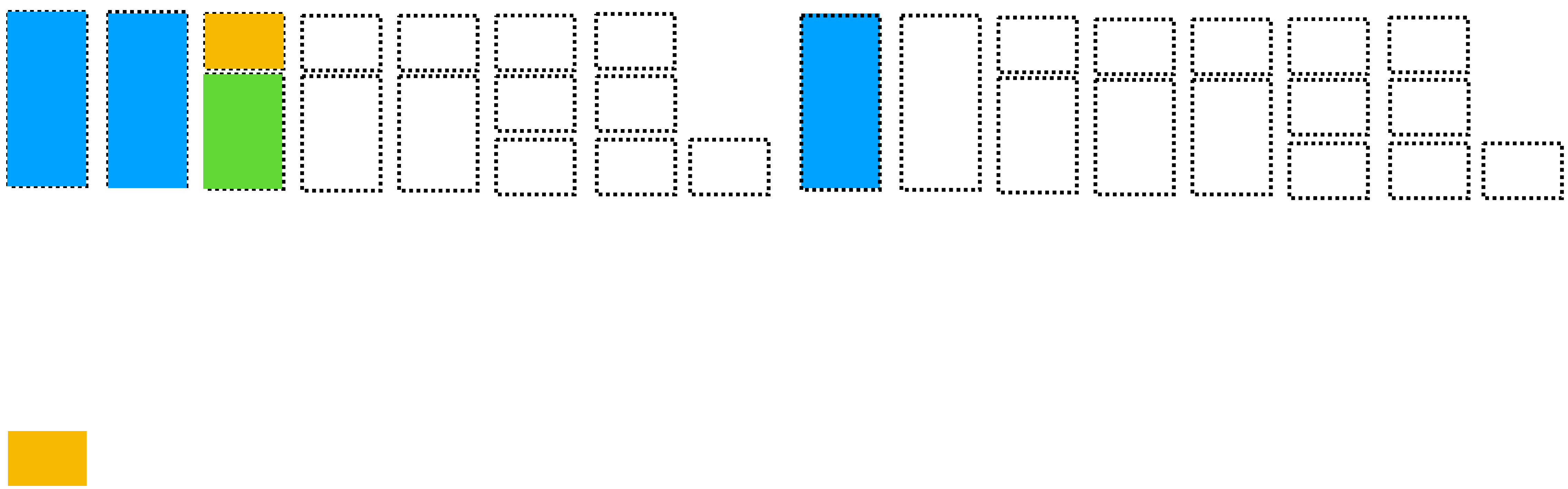


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

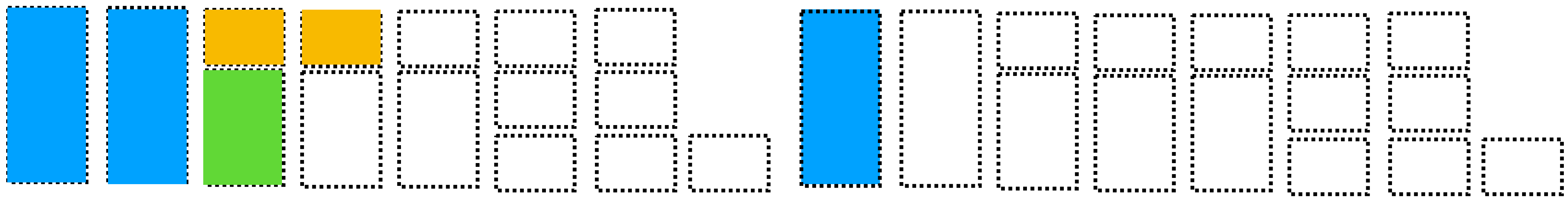


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

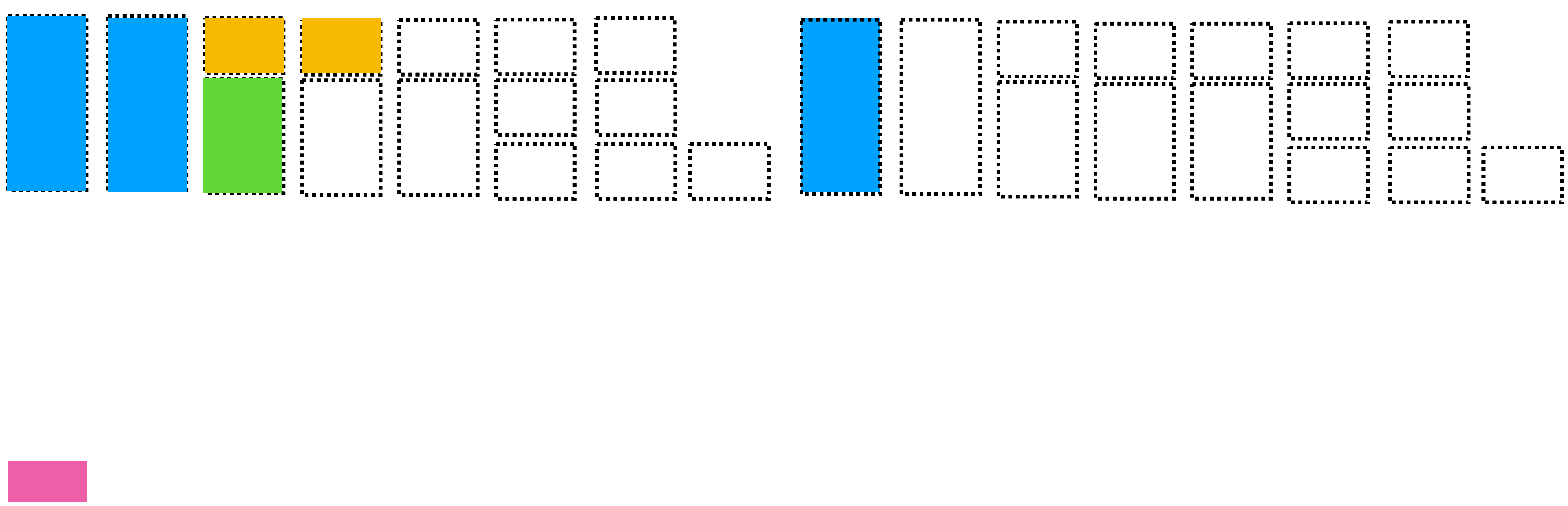


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

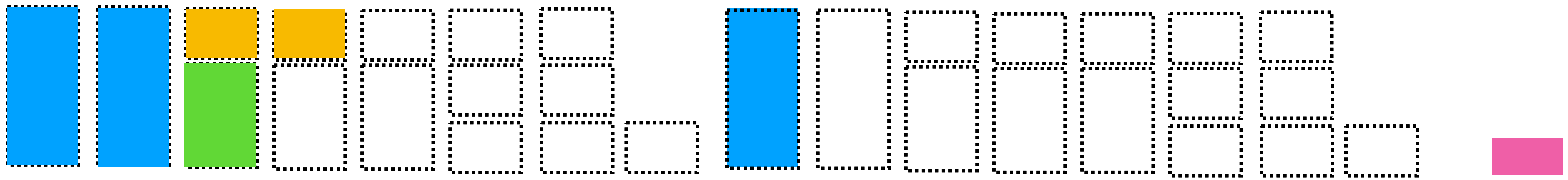


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

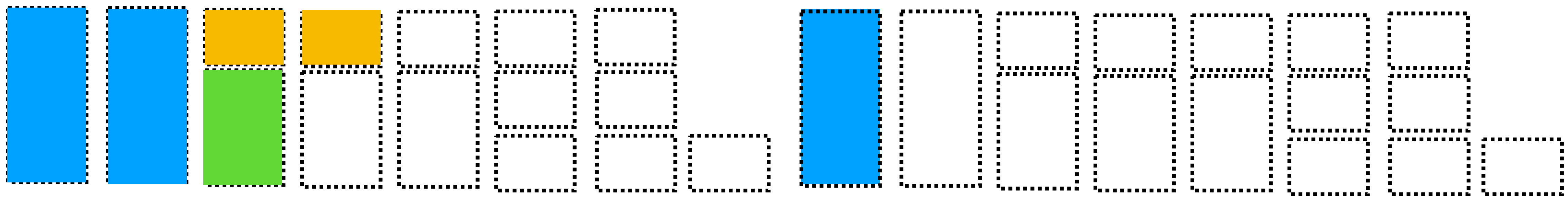


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

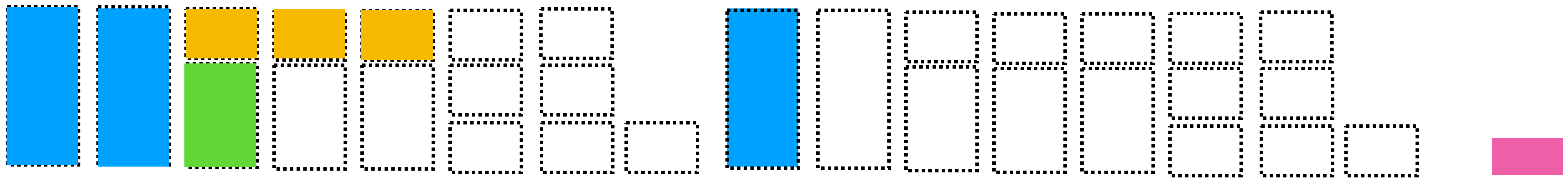


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

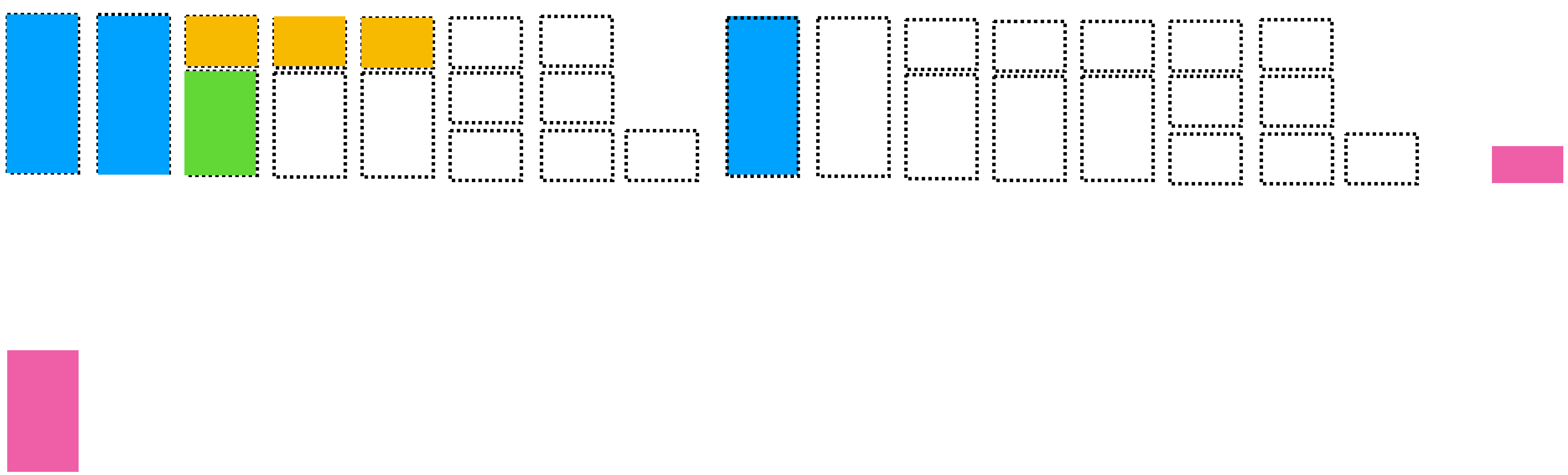


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT

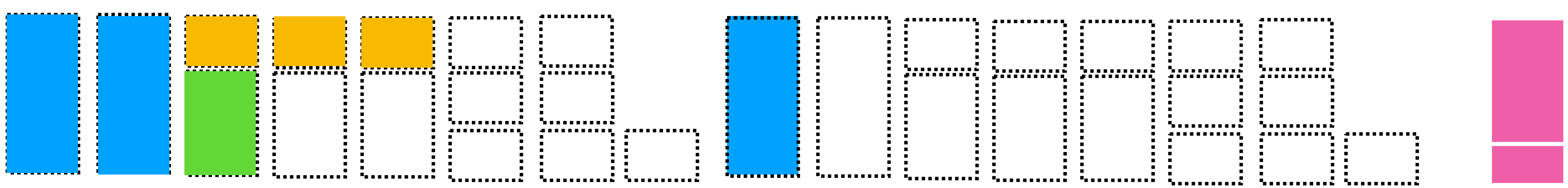


Profile packing

Main idea: Pack the items by placing them in their profile placeholder

Algorithm opens profiles instead of simple bins (virtually)

Items that were not predicted to appear are packed separately using FIRST-FIT



Some main results

Some main results

Theorem: Profile Packing has consistency $1 + \epsilon$, and competitive ratio at most

$$1 + (2 + 5\epsilon)\eta k + \epsilon$$

(excellent consistency, bad robustness)

Some main results

Theorem: Profile Packing has consistency $1 + \epsilon$, and competitive ratio at most

$$1 + (2 + 5\epsilon)\eta k + \epsilon$$

(excellent consistency, bad robustness)

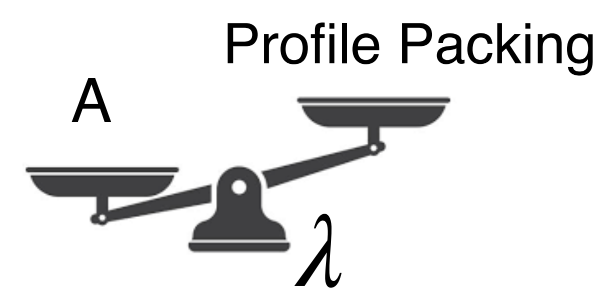
Lower bound: Let $c < 1$ be a constant. For any $\alpha \leq c/k$, any algorithm that is $(1 + \alpha)$-consistent must have robustness at least $(1 - c)k/2$

Robustification

Robustification

We define an algorithm **HYBRID**(λ)**,** where $\lambda \in [0,1]$ is a parameter chosen by the user

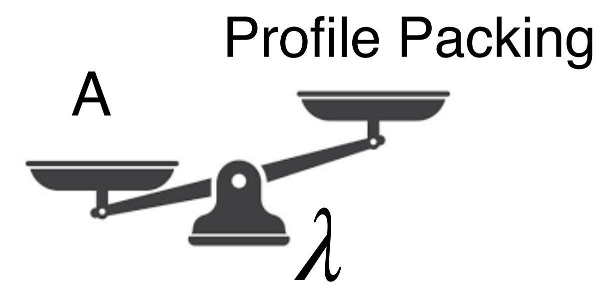
Main idea: Some items are served using a competitive algorithm (A), and others using Profile Packing



Robustification

We define an algorithm **HYBRID**(λ), where $\lambda \in [0,1]$ is a parameter chosen by the user

Main idea: Some items are served using a competitive algorithm (A), and others using Profile Packing



Algorithm :

— Keep track of the number of items served by PP, and the total number of items

Specifically: counters $ppcount(x)$ and $count(x)$ for all $x \in [1,k]$

— Upon arrival of an item of size x

- If there is an available placeholder, use it, and declare it a PP-item
- Otherwise, if $ppcount(x) \leq \lambda \cdot count(x)$, serve it using PP

  Else serve it using A

Robustification (results)

Theorem: HYBRID(λ) has competitive ratio at most

$$(1+\epsilon)((1+(2+5\epsilon)\eta k)\lambda + c_A(1-\lambda))$$, where $c_A =$ comp. ratio of A

Better results can be achieved assuming knowing some upper bound on the error

Robustification (results)

Theorem: HYBRID(λ) has competitive ratio at most

$$(1+\epsilon)((1+(2+5\epsilon)\eta k)\lambda + c_A(1-\lambda)),$$ where c_A = comp. ratio of A

Better results can be achieved assuming knowing some upper bound on the error

Note: Some approaches that do not work:

— Skip the first step of the algorithm

— Algorithms along the lines of [Mahdian, Nazerzadeh and Saberi 2012]

Extensions

- Improvements when few items can be packed in a bin (VM placement)
- Sampling-based randomized online algorithms
- Handling fractional items

Handling fractional items

Handling fractional items

If a fractional item appears, serve it separately using First Fit

We need a measure of the “integrality" of a sequence, e.g., $d(\sigma) = \sum_{x \in \sigma} |x - \lfloor x \rceil|$

This measure is too restrictive: no online algorithm with frequency predictions

can have competitive ratio better than 4/3, even if $\eta = 0$, and $d(\sigma) = \epsilon$

Handling fractional items

If a fractional item appears, serve it separately using First Fit

We need a measure of the “integrality" of a sequence, e.g., $d(\sigma) = \sum_{x \in \sigma} | x - \lfloor x \rceil |$

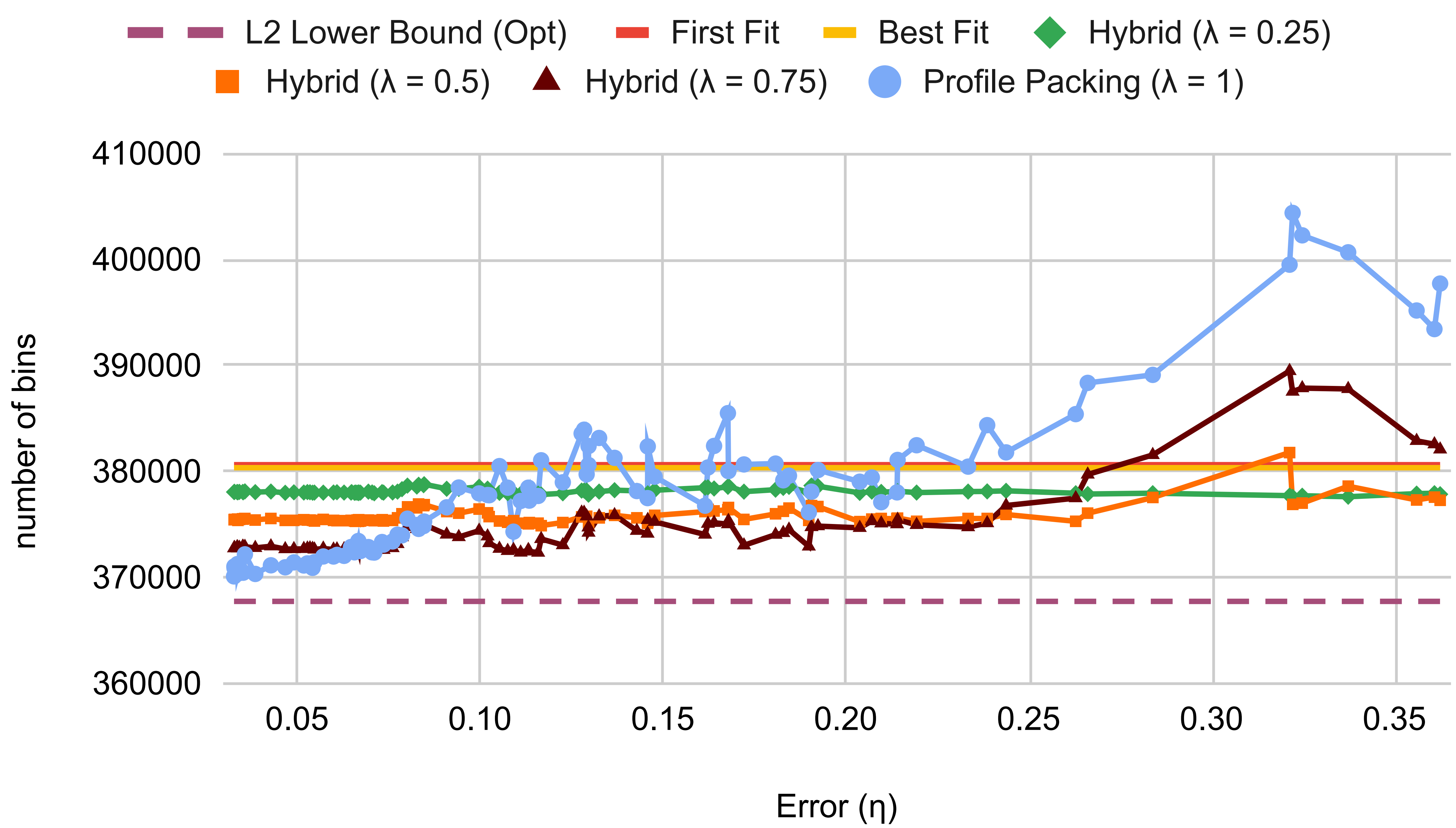
This measure is too restrictive: no online algorithm with frequency predictions can have competitive ratio better than 4/3, even if $\eta = 0$, and $d(\sigma) = \epsilon$

Alternative: $\hat{d}(\sigma) = \frac{\sum_{x \in \sigma, x \neq \lfloor x \rceil} x}{\sum_{x \in \sigma} x}$ (ratio of “fractional” sizes over total sizes)

Result: If an algorithm with frequency predictions has competitive ratio c for integral sizes, then we can transform it to an algorithm of competitive ratio $c + 2\hat{d}(\sigma)$ for fractional sizes

Experimental analysis

Weibull Fixed



Future work

- Further improve the lower bounds
- Multi-dimensional bin packing
- Extend to full (i.e., “continuous”) model (caveat: advice complexity impossibility results)
- Evolving distributions

Future work

- Further improve the lower bounds
- Multi-dimensional bin packing
- Extend to full (i.e., “continuous”) model (caveat: advice complexity impossibility results)
- Evolving distributions

Full paper available at www.jair.org

Future work

- Further improve the lower bounds
- Multi-dimensional bin packing
- Extend to full (i.e., “continuous”) model (caveat: advice complexity impossibility results)
- Evolving distributions

Full paper available at www.jair.org

Thank you!

A sampling-based online algorithm

We can use the PAC-learnability of frequency predictions to obtain a randomized algorithm that mixes a robust algorithm A and Profile Packing

Result: For any $\epsilon > 0$, there is a randomized algorithm with s samples that has expected competitive ratio $(1-\delta)(1+\epsilon)((1+(2+5\epsilon)\eta k+\epsilon)+c_A\delta$, where $\delta = 1/\sqrt{2^{s\eta^2-k}}$