Improved Approximation for Vector Bin Packing

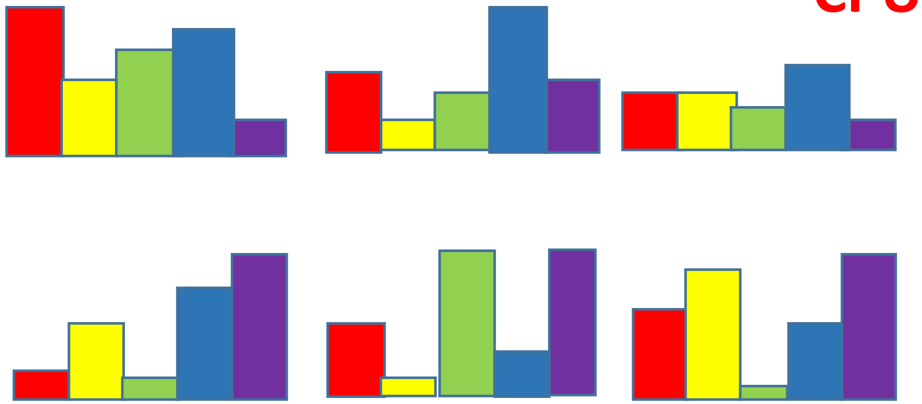
Arindam Khan

(Georgia Tech → IDSIA, Lugano, Switzerland)

(Joint work with Nikhil Bansal and Marek Elias at TU Eindhoven)

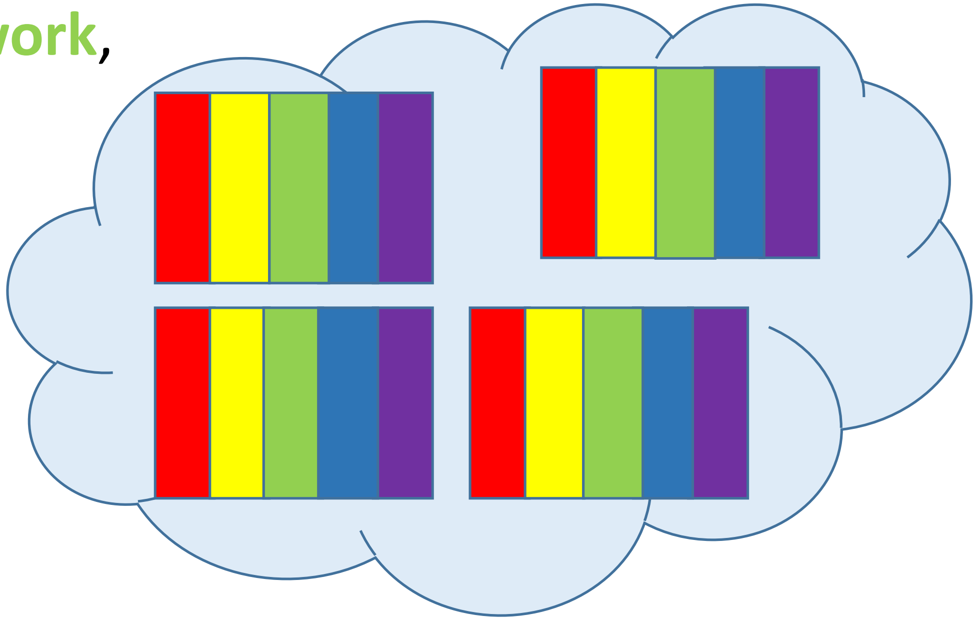
Vector Packing: Multidimensional Bin Packing

Jobs (items)



Resources (dimensions):
CPU, **Memory**, **Network**,
Disk, **I/O**

Servers (bins)

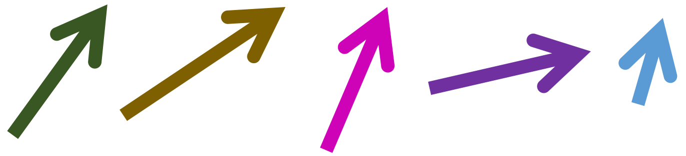


Goal: Assign all jobs to the servers s.t. min number of servers are needed.

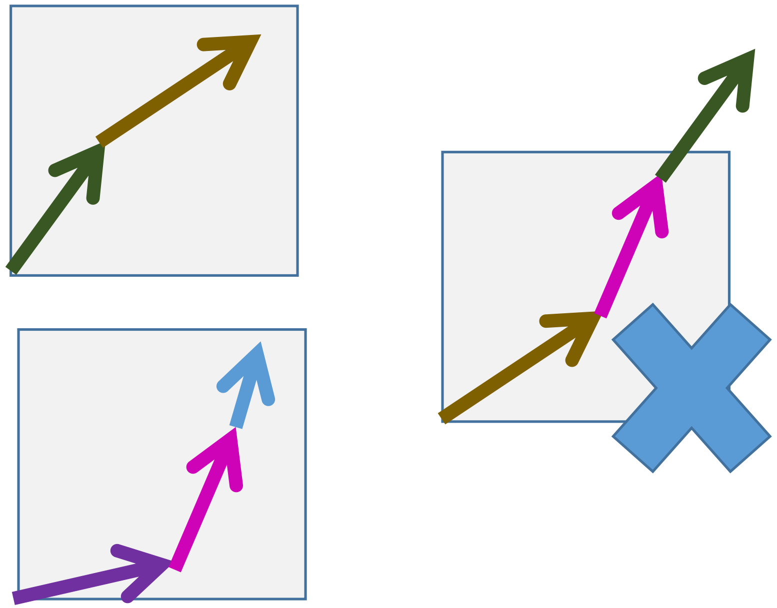
Vector packing

- Input:

Set of d-dimensional vectors: $[0,1]^d$



$d = 2$



- Goal:

pack all vectors into minimum # of unit vector bins such that for each bin for each dimension the coordinate wise sum of packed vector in it is $\leq\ 1$.

Applications:

- Classical Generalization of Bin Packing ($d = 1$).
- Multi processor scheduling
  -- Cloud computing.
- Layout Design.
- Multi-objective resource allocation.
- Logistics and loading problems.

Asymptotic Approximation:

- Even for 1-D Bin Packing: NP Hardness from *Partition*
  - Can't distinguish in poly time if need 2 or 3 bins.
  - 3/2 hardness when OPT=2.
- What happens when OPT is large?
- Asymptotic Approximation Ratio is ρ
  $if\ Algo(I) \leq \rho . OPT(I) + O(1).$
- Asymptotic Polynomial Time Approximation Scheme (APTAS):
  $if\ Algo(I) \leq (1+\epsilon)\, Opt(I) + O_{\epsilon}(1)$

Vector Packing: a tale of approximability

- Dimension d is constant.
- Asymptotic Approximation:
  - $d + \epsilon$ [Linear grouping: de la Vega-Lueker '81]
  - $2 + \ln(d) + \epsilon$ [Assignment LP: Chekuri-Khanna '99]
  - $1 + \ln(d) + \epsilon$ [Configuration LP: Bansal-Caprara-Sviridenko FOCS '06]
- Absolute/Nonasymptotic: 2 for $d = 2$ [Kellerer-Kotov 2003]
- Hardness:
- No APTAS (from 3D Matching)[Woeginger 1997].

Our Results:

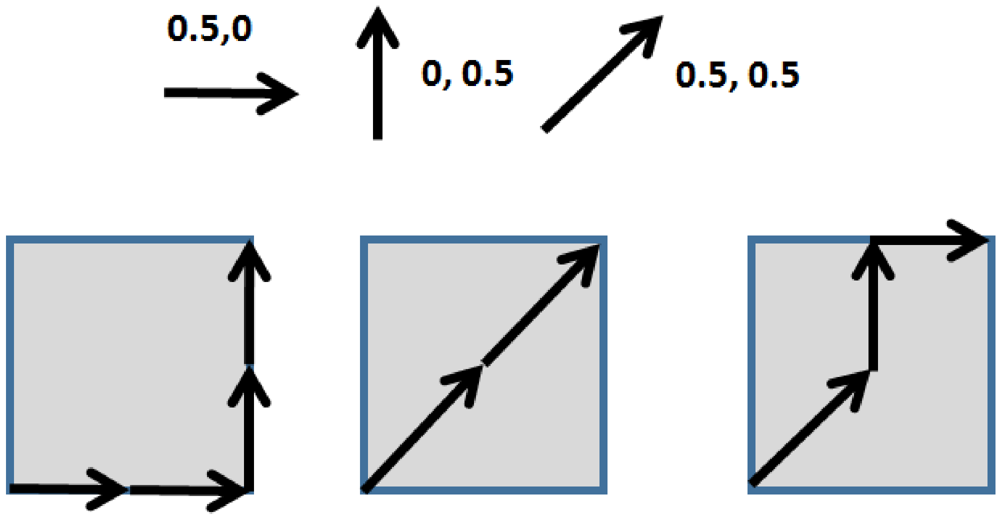
- Algorithm:
- Almost tight $(1.5 + \epsilon)$ (Absolute) Approximation for 2-D. (Prev: 2)
- 1.405 Asymptotic Approximation for 2-D. (Prev: 1.69)
- $0.807 + \ln(d+1)$ Asymptotic Approximation for d dim. (Prev: $1 + \ln d$)
- Hardness of d for constant rounding based algorithms.
- Resource Augmentation:
- If we allow extra resource of ϵ in $(d-1)$ dimensions, we can find a packing in polynomial time in $(1+\epsilon)Opt + O(1)$ number of bins.

Bin Packing LP Relaxation : Configuration LP

- *$\mathcal{B}$: set of configurations (possible way of feasibly packing a bin).*

- $\min \{ \sum_C x_C \; : \; \sum_{C \ni i} x_C \geq 1 \; (i \in I), x_C \in \{0,1\} \; (C \in \mathcal{B}) \}$

- Objective: Minimize #selected configurations (bins).
- Constraint: For each item, at least one configuration containing the item should be selected. (packing all items)



- Problem: Exponential # variables!.
- Solution: $(1+\epsilon)$approx. of vector knapsack (separation problem of dual.)

Round & Approx Framework (R & A) [Bansal-Caprara-Sviridenko '06]

- 1. Solve configuration LP.
- 2. Randomized Rounding: For few $(??)$ iterations :

  select a configuration C' at random with probability $\frac{x^*_{C'}}{LP(I)}$.

- 3. Approx: Let S be the set of remaining uncovered elements. Pack S using a $(d+\epsilon)$ approximation algorithm.

  [Replace each item v by $||v||_\infty$ and pack using 1-D bin packing.]

- This gives $(1+\ln d)$ approximation by choosing few = $\ln d \cdot LP(I)$.

Round & Approx Framework (R & A) Extended [Bansal-K. SODA'14]

- Let $\mathcal{A}$ be a ρ-approximation algorithm where all items are rounded to O(1) number of values.

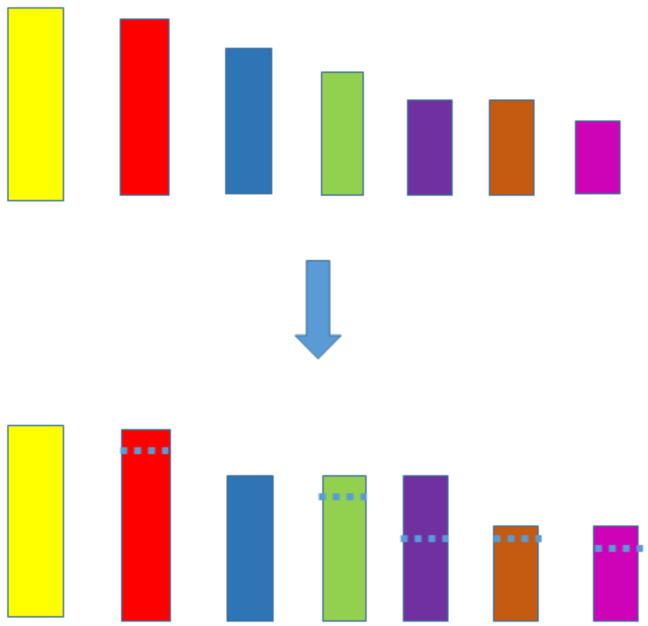
- 1. Solve configuration LP.
- 2. Randomized Rounding: For $\ln \rho \cdot LP(I)$ iterations :

  select a configuration C' at random with probability $\frac{x^*_{C'}}{LP(I)}$.
- 3. Approx: Each item is left with probability $\frac{1}{\rho}$.

  Pack remaining uncovered elements using $\mathcal{A}$.
- This gives $(1 + \ln \rho)$ approximation.

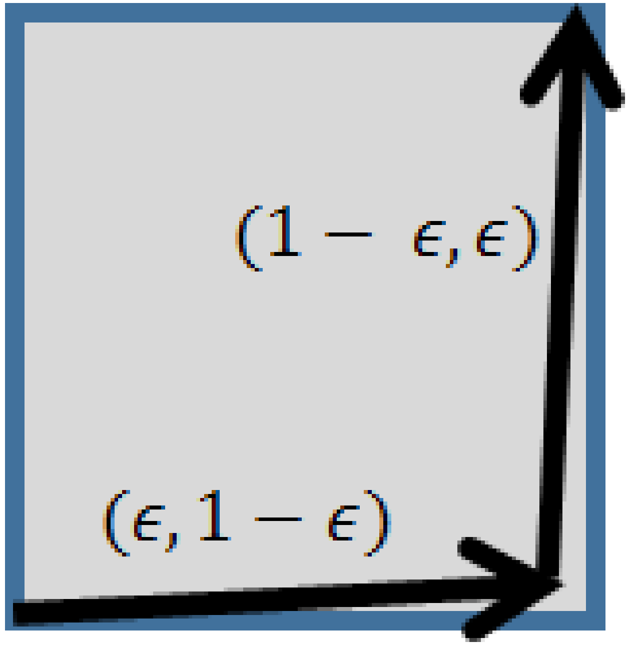
Rounding based Algorithms:

- Ubiquitous in bin packing: Linear grouping, Geometric Grouping [Karp-Karmarkar], Harmonic Rounding [Lee-Lee.]
- Large items are replaced by larger items of O(1) types.
- Loss: Due to larger items.
- Gain: Fewer configurations.
- Theorem: d-approximation is tight for algorithms that round the large coordinates to $O(1)$ number of values.
- $(1 + \ln d)$ approximation is tight by R&A framework.
- How to break this natural barrier of $1 + \ln d$?.

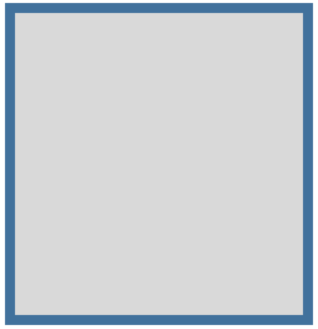


Beating $(1 + \ln d)$

- Any better approx. algorithm must implicitly or explicitly consider the original (unrounded) sizes of items while packing them
- Consider $d = 2$
- Tight example for rounding based algorithms: When there are only two items u, v in the bin with $u + v \geq (1 - \epsilon, 1 - \epsilon)$.-- matching bins.
- If no such bins we get a 3/2 approximation using a structural lemma along with resource augmentation.

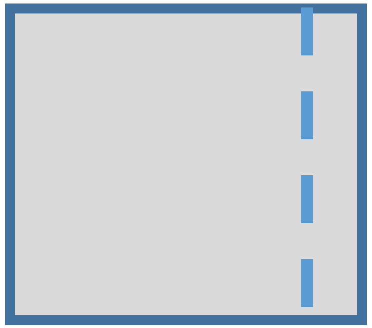


Resource Augmentation



- Allow extra resource ϵ in one dimension.

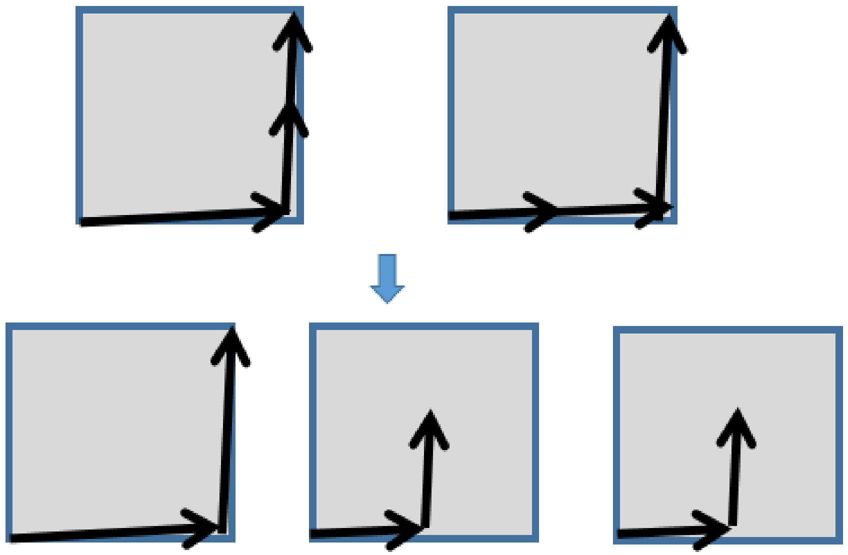
- Theorem:
  If we allow resource augmentation in $(d-1)$ dimensions we can pack items in $(1+\epsilon)Opt$ number of bins.



- Round big items (linear grouping) in non-augmented dim.,
  Round up items to multiple of ϵ^2 in other dim.
  $\Rightarrow$ O(1) types of big coordinates.
- Find optimal packing of rounded big items.
- Pack small items using an assignment Linear Program.

Structural lemma

- 2-D vector packing: Any packing of m bins can be transformed into a packing of $3m/2$ bins where each bin either contains 2 items or has slack in one of the dimensions.
- (Existential Result) A packing of $\approx 3m/2$ bins where each bin either contains 2 items (matching bins) or has O(1) types of big items (nonmatching bins).



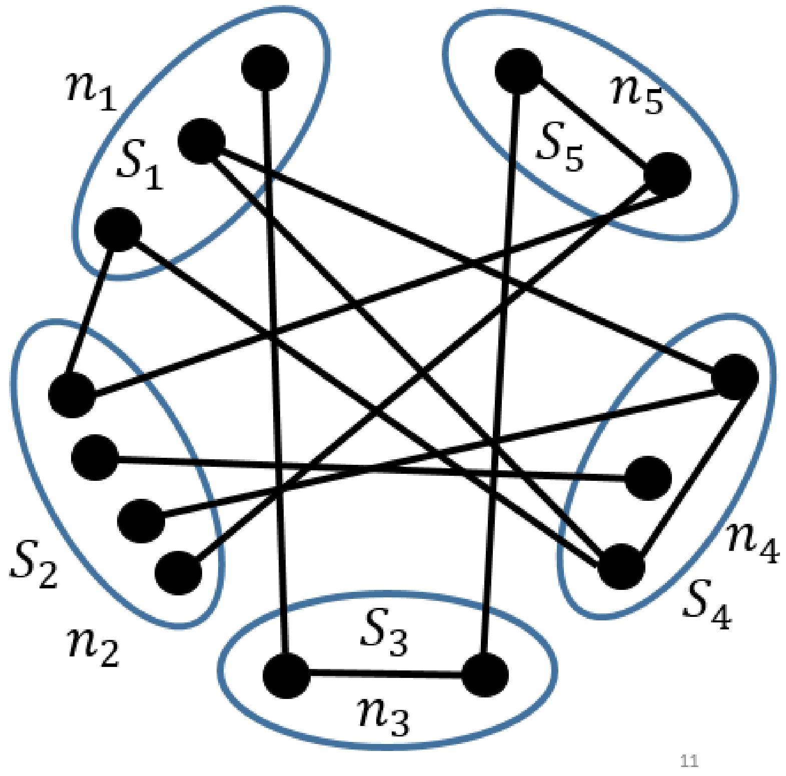
- d-Dimensions: Any packing of m bins can be transformed into a packing of packing of $\approx 2m$ bins where m bins contain $\leq\ d-1$ items (compact bins) in it and other $\approx m$ bins have slack in $(d-1)$ dimensions (noncompact bins).

2 D vector packing

- Matching bins: create a graph with nodes = items in matching bins, Edge (u, v) if the items u and v can be packed into one bin. – Pack using matching!
- Nonmatching bins: packing is based on O(1) types of rounded items. We can find the rounding specification (i.e., rounded values and number of items in each size class) in polynomial ($n^{O(1)}$) time.
  – Rounded specification is sufficient, does not need original item sizes.
- If we can separate out items in matching bins and nonmatching bins, we can pack each of them separately.
- However we don't know which items are packed in which bins!

MultiObjective MultiBudget Matching
[Chekuri-Vondrak-Zenklusen SODA'11]

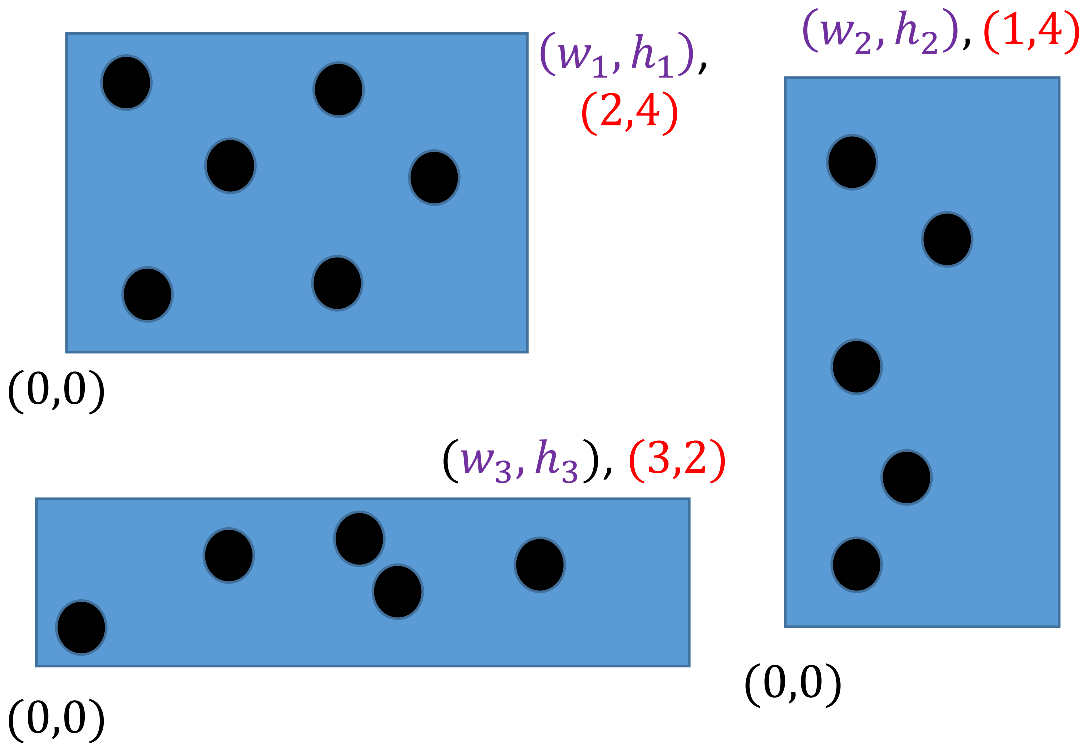
- Given a graph and a partition of its vertices s.t. $V := S_1 \cup S_2 \cup \cdots S_k$ and numbers $n_1, n_2, \cdots n_k$; there is a poly time Algorithm that finds a matching (if exists) that saturates *nearly* n_i items from each S_i.
- Vector Packing:
  - Nodes (V): items
  - S_i: sizeclasses,
  - Edge (u, v) if the items u and v can be packed into one bin.



2-D : Overview of 3/2 Approximation

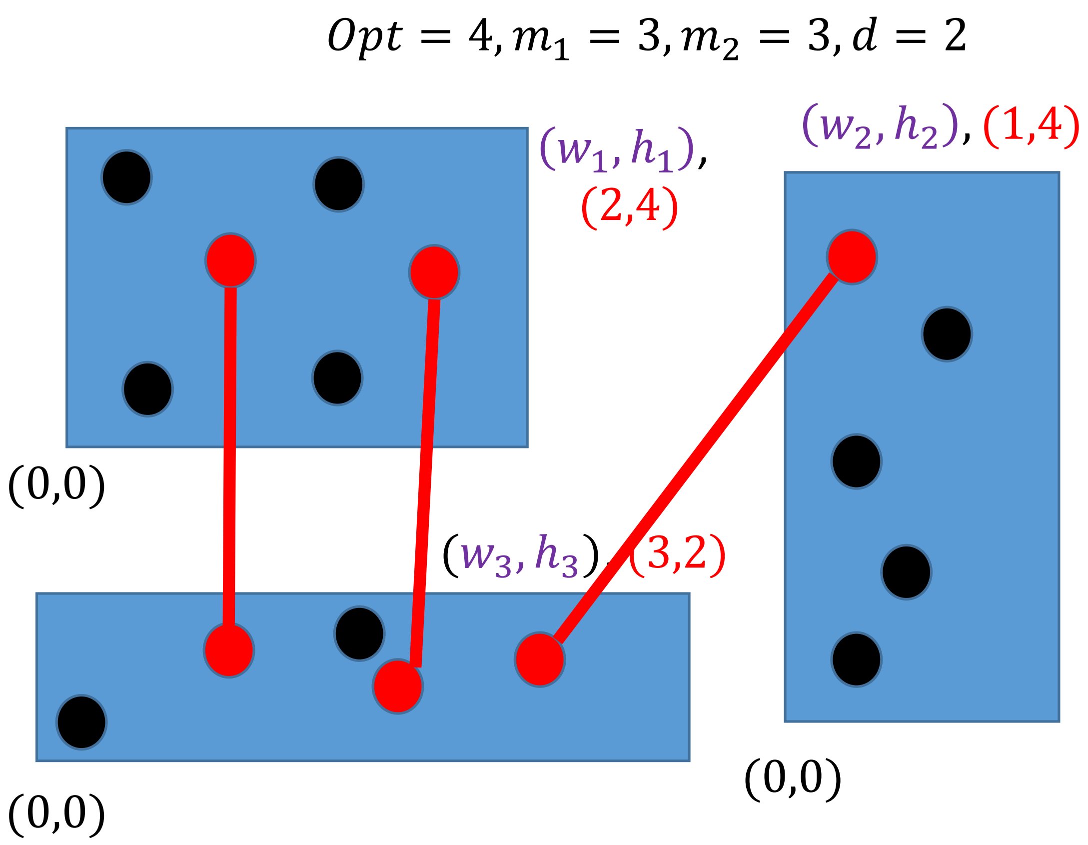
- Guess Opt, num. of matching bins m_1, num. of nonmatching bins m_2 where $m_1 + m_2 \leq \frac{3}{2} Opt$.
- Guess $O(1)$ types of **rounded size classes** and **number of items** in each size class in matching and nonmatching bins. (items in matching bins are not rounded, they are just assigned to classes)

$Opt = 4, m_1 = 3, m_2 = 3, d = 2$



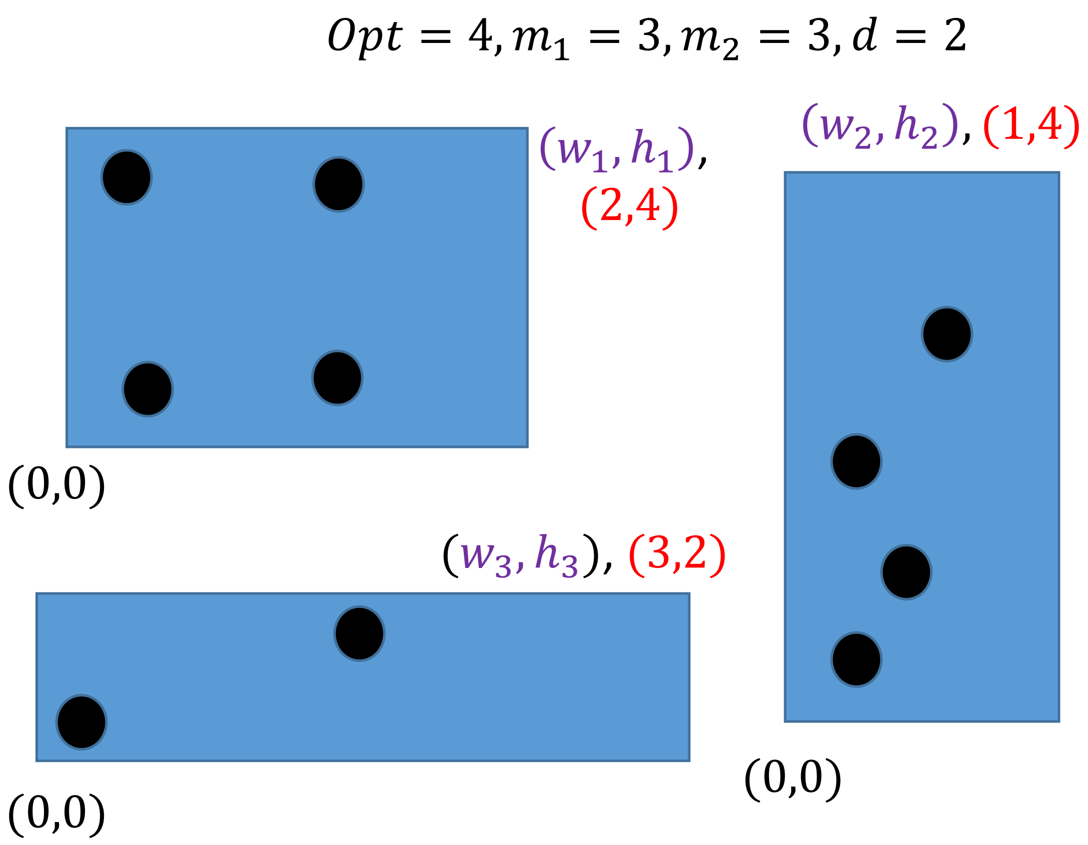
2-D : Overview of 3/2 Approximation

- Guess Opt, num. of matching bins m_1, num. of nonmatching bins m_2 where $m_1 + m_2 \leq \frac{3}{2} Opt$.
- Guess $O(1)$ types of **rounded size classes** and **number of items** in each size class in matching and nonmatching bins. (items in matching bins are not rounded, they are just assigned to classes)
- Use multi-objective matching using original sizes to pack items into $(1+\epsilon)m_1$ bins.



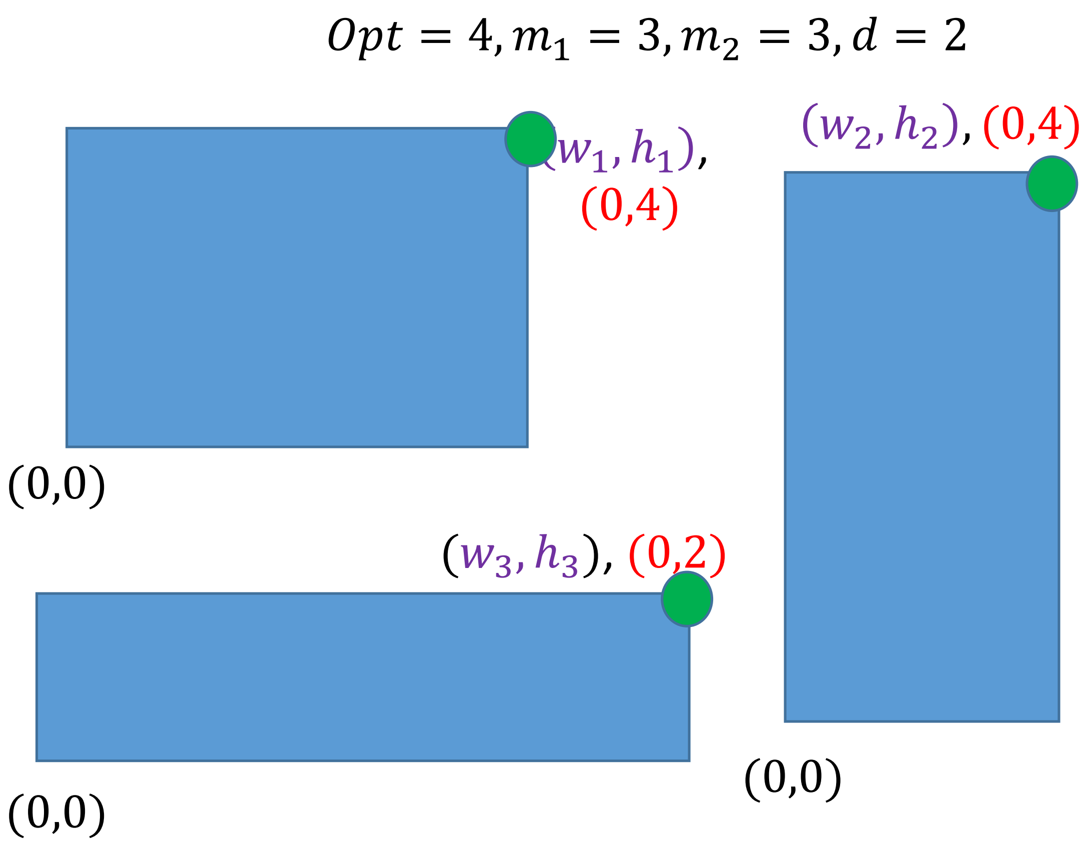
2-D : Overview of 3/2 Approximation

- Guess Opt, num. of matching bins m_1, num. of nonmatching bins m_2 where $m_1 + m_2 \leq \frac{3}{2} Opt$.
- Guess $O(1)$ types of **rounded size classes** and **number of items** in each size class in matching and nonmatching bins. (items in matching bins are not rounded, they are just assigned to classes)
- Use multi-objective matching using original sizes to pack items into $(1 + \epsilon)m_1$ bins.



2-D : Overview of 3/2 Approximation

- Guess Opt, num. of matching bins m_1, num. of nonmatching bins m_2 where $m_1+m_2 \leq \frac{3}{2} Opt$.
- Guess $O(1)$ types of **rounded size classes** and **number of items** in each size class in matching and nonmatching bins. (items in matching bins are not rounded, they are just assigned to classes)
- Use multi-objective matching using original sizes to pack items into $(1+\epsilon)m_1$ bins.
- Use rounding based algorithm to pack remaining items into $(1+\epsilon)m_2$ bins.



R & A Framework beyond $(1 + \ln d)$

- More technical !
- 2-D:
  Matching $\rightarrow$ Rand. Rounding $\rightarrow$ Rounding based algo.
- $1 + \ln(1.5) \approx 1.405$ -approximation.
- d-D: No such theorem for multiobjective d-D matching!

R & A Framework beyond $(1 + \ln d)$

- d-Dimensions:
- Random Rounding $\rightarrow$ Matching $\rightarrow$ Rounding based algo.
- Compact bins contain $\approx dm$ items.
- After random rounding $\approx 2m$ items are left from compact bins.
- Now we use multiobjective matching to pack in $\approx 1.5m$ bins.
- $1.5 + ln\left(\frac{d+1}{2}\right)$ -approximation. ■

Open Problems!

- Improved approximation ($\ln \ln d?$) or inapproximability (as $f(d)$)
- Understanding the integrality gap of configuration LP.
- Generalize multiobjective matching to higher dimensions > 2
  - Can give better approximation for small dimensions.
  - Can not beat $O(\ln d)$ by our approach.
- Other generalizations of bin packing (geometric bin packing, geometric knapsack, strip packing, weighted bipartite edge coloring) – Read my Thesis!

Questions!

Extra Slides

Arindam Khan (Georgia Tech → IDSIA, Lugano, Switzerland)

(Joint work with Nikhil Bansal and Marek Elias at TU Eindhoven)

Configuration LP

- $\mathbb{C}$*: set of configurations(possible way of feasibly packing a bin).*

Primal: **LP(I)**

$$\min \{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \ (i \in I), x_C \geq 0 \ (C \in \mathbb{C}) \}$$

Dual:

$$\max \{ \sum_{i \in I} v_i : \sum_{i \in C} v_i \leq 1 \ (C \in \mathbb{C}), v_i \geq 0 (i \in I) \}$$

Dual Separation problem =>
d-D Vector Knapsack problem:

$$\sum_{i \in C} v_i^* > 1 \ (for\ some\ C \in \mathbb{C})$$

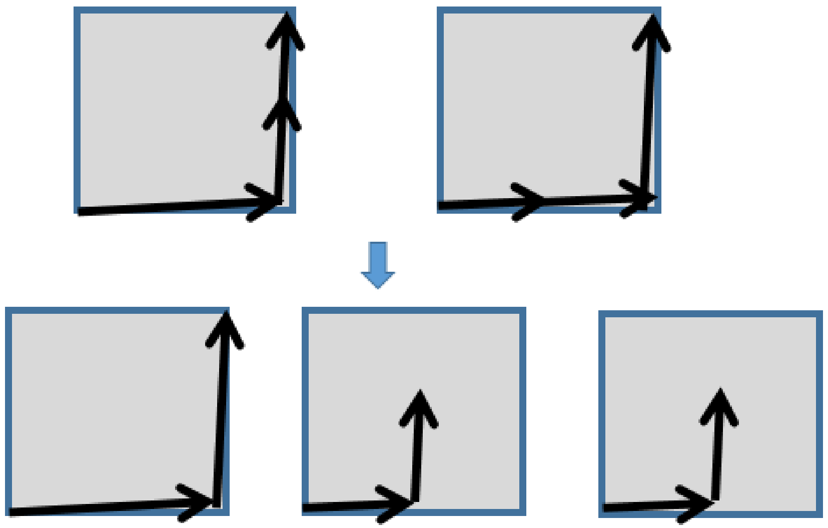
- Max $\sum_{\{i \in I\}} v_i^* x_i$
- S.t. $\sum_{\{i \in I\}} s_i^k x_i \leq 1$, for $k \in [d]$.

$$x_i \in \{0,1\}$$

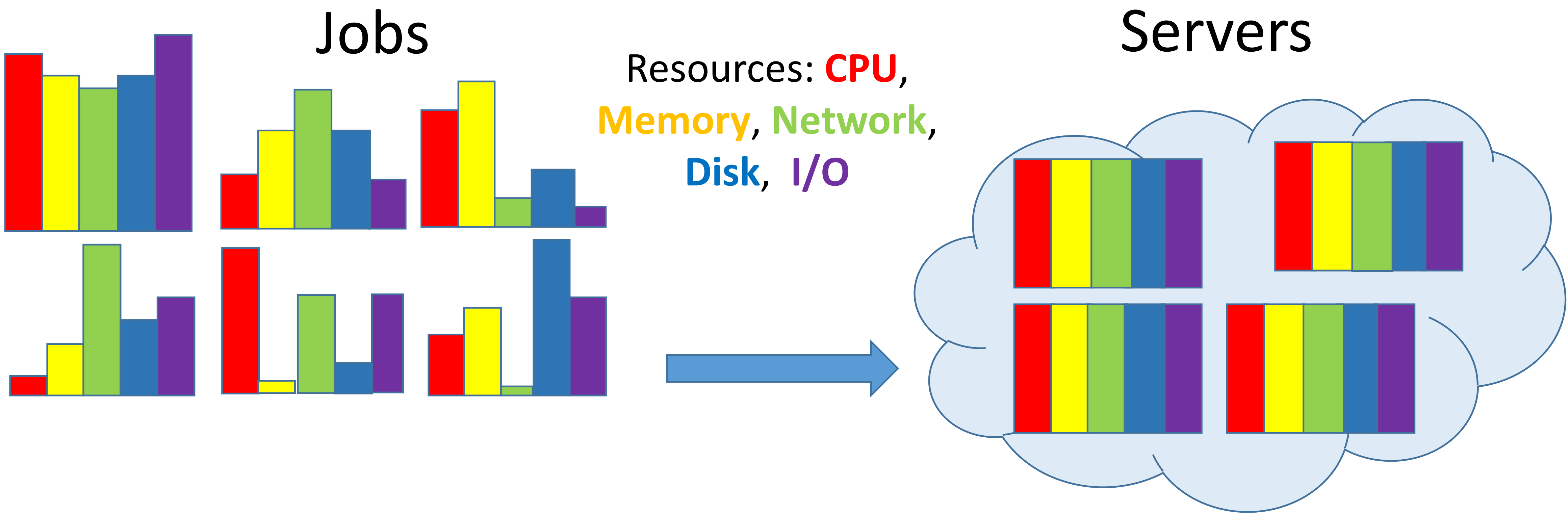
- Problem: Exponential number of configurations!
- Solution: Can be solved within $(1+\epsilon)$ accuracy using separation problem for the dual. [Frieze-Clarke '84]

Proof Sketch (2 Bins → 3 match./nonmatch. bins)

- 2 D vector packing: Small items: $< (\epsilon, \epsilon)$, Big Items: Otherwise.
- If not matching → ≥ 3 items
- Each dim, only one item $>$ ½.
- One item $\leq \left(\frac{1}{2}, \frac{1}{2}\right)$ in each bin [candidate item]
- Two candidate items c_A, c_B can be packed into one matching bin D.
- If candidate item is big, we get enough slack.
- Otherwise, we can remove a subset of small items from A (and B) to get bins E, F with slack and pack removed items in D.

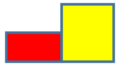


Vector Packing: Multidimensional Bin packing



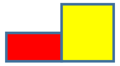
Goal: To assign all jobs to the servers such that minimum number of servers are needed.

Vector packing (Two Dimensions)

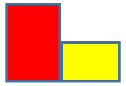


(0.3, 0.5)

(0.5, 0.1)



(0.25, 0.5)

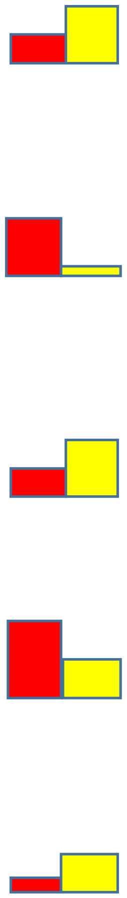


(0.6, 0.4)

(0.1, 0.3)

Vector packing: (Two Dimension)

Jobs/Items



(0.3, 0.5)

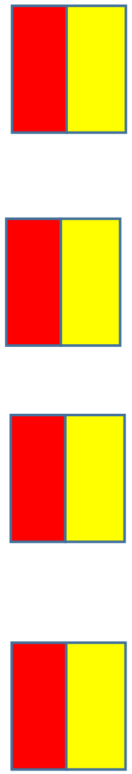
(0.5, 0.1)

(0.25, 0.5)

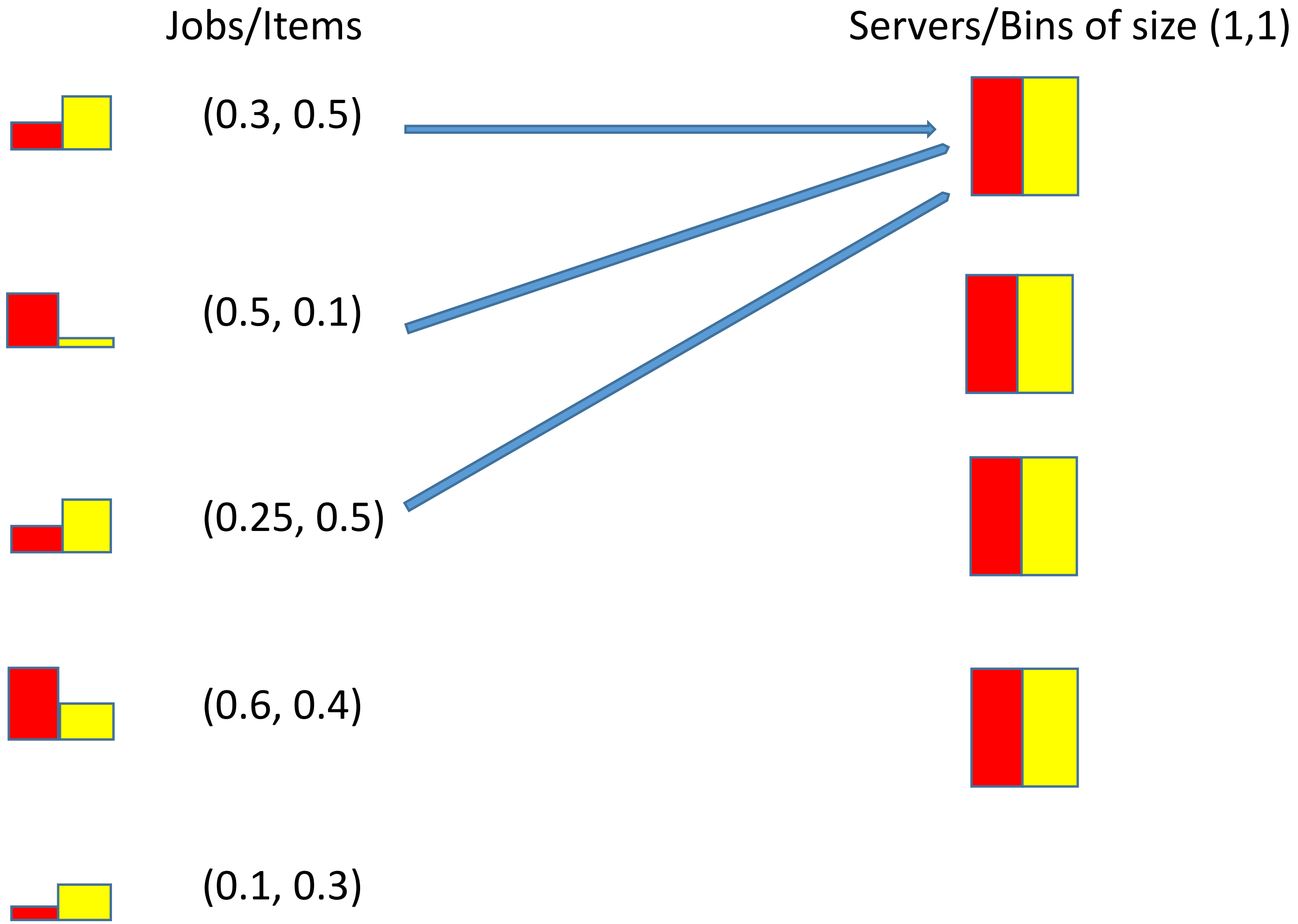
(0.6, 0.4)

(0.1, 0.3)

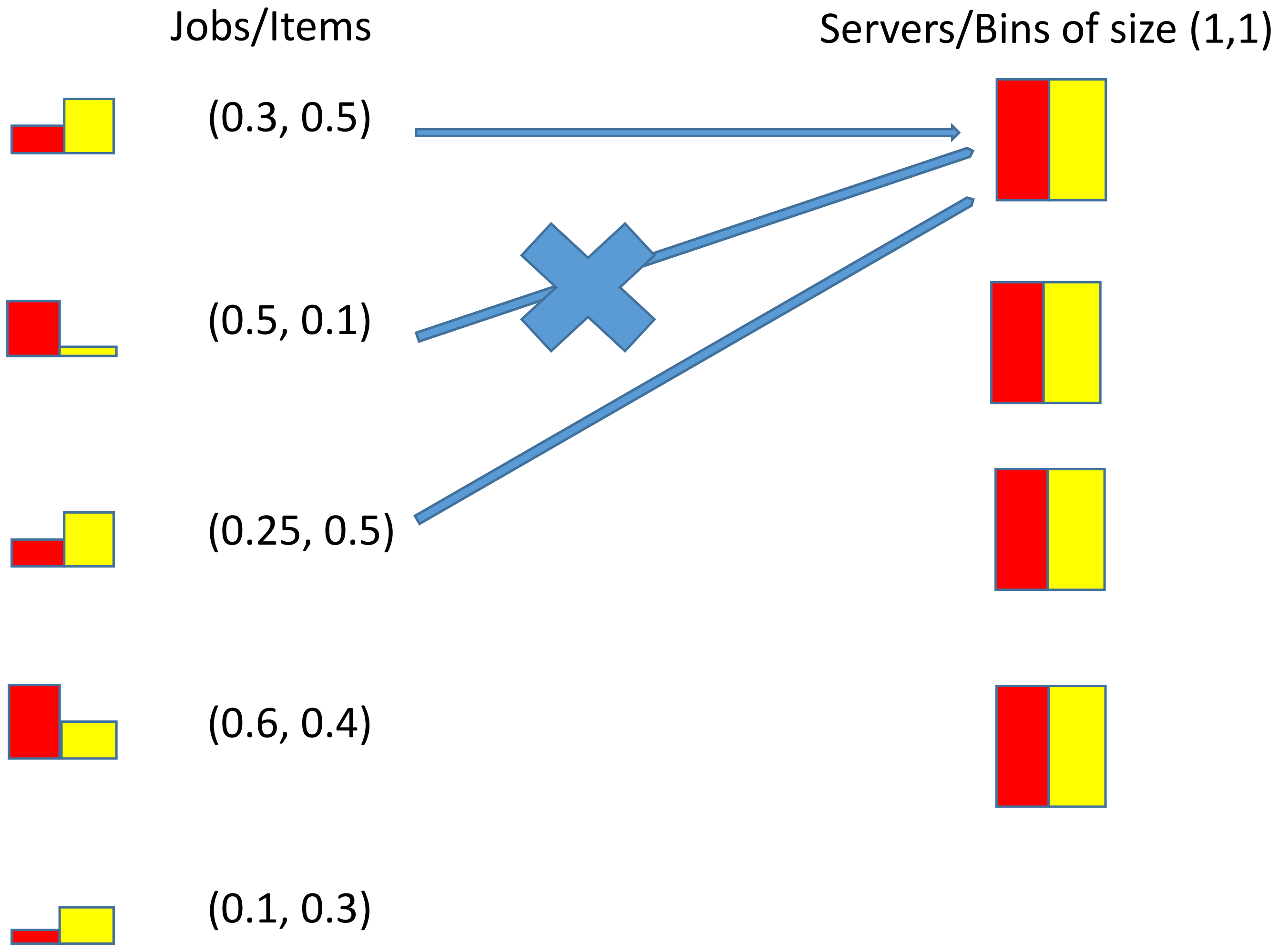
Servers/Bins of size (1,1)



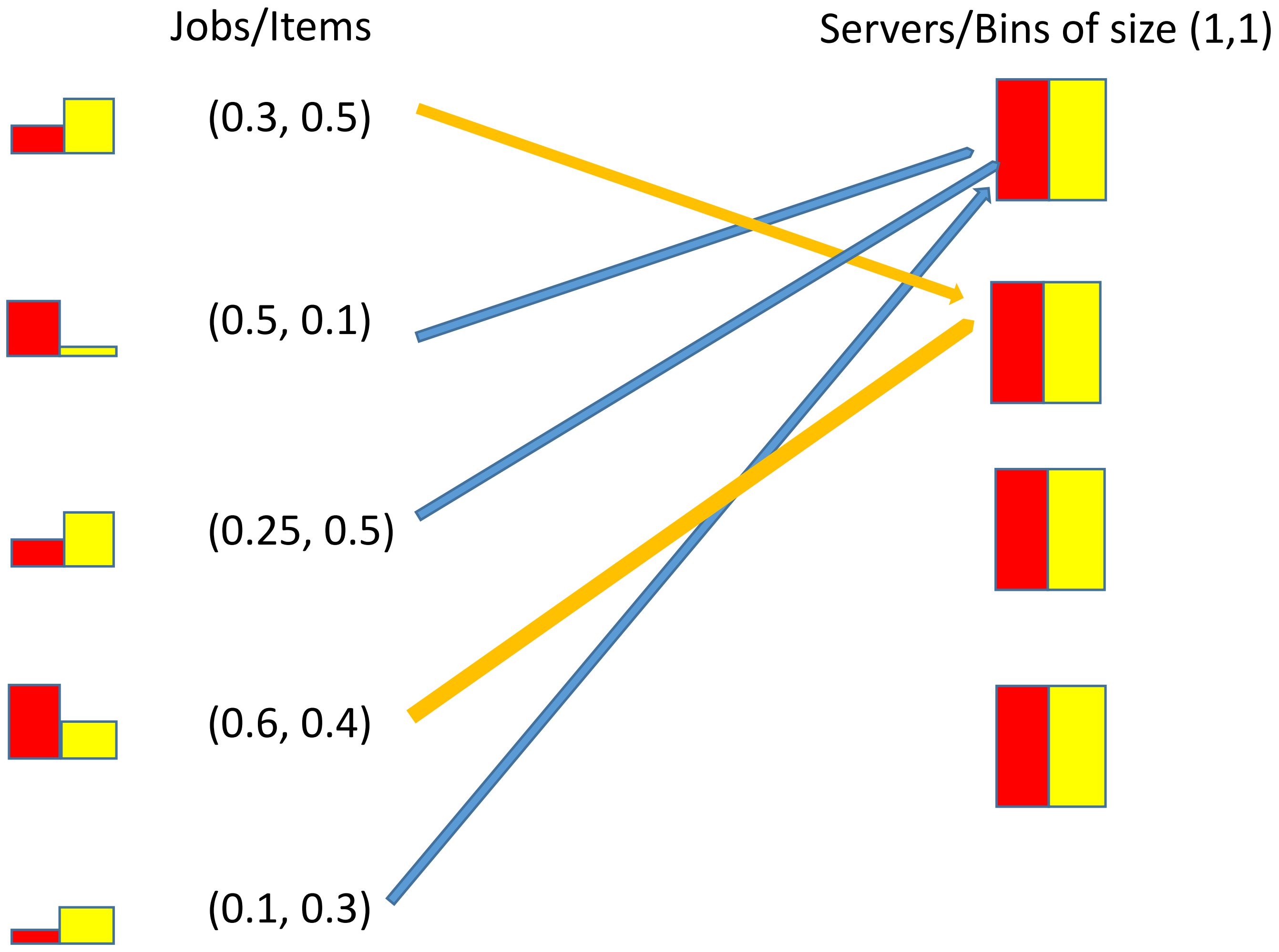
Vector packing: (Two Dimension)



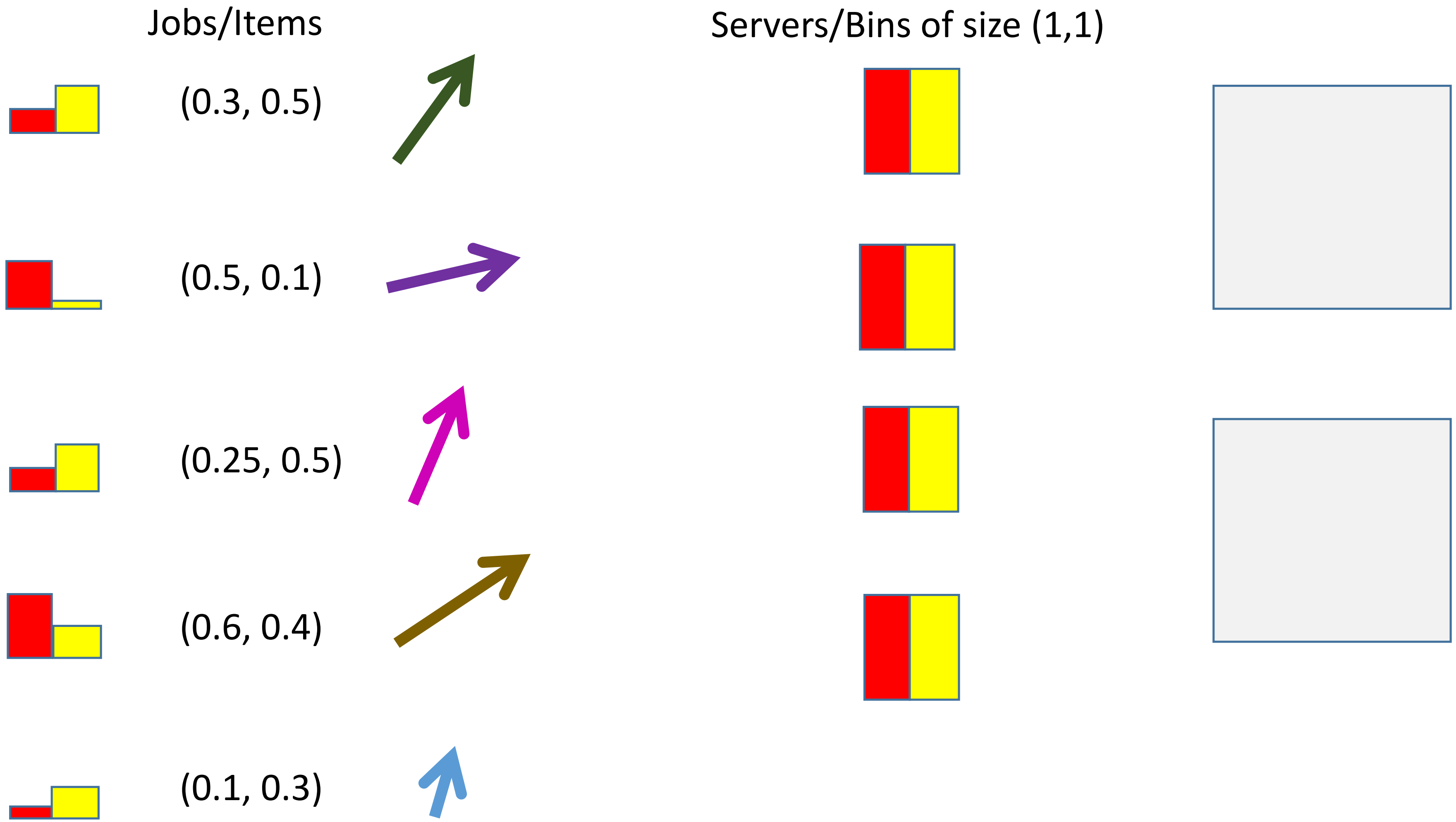
Vector packing: (Two Dimension)



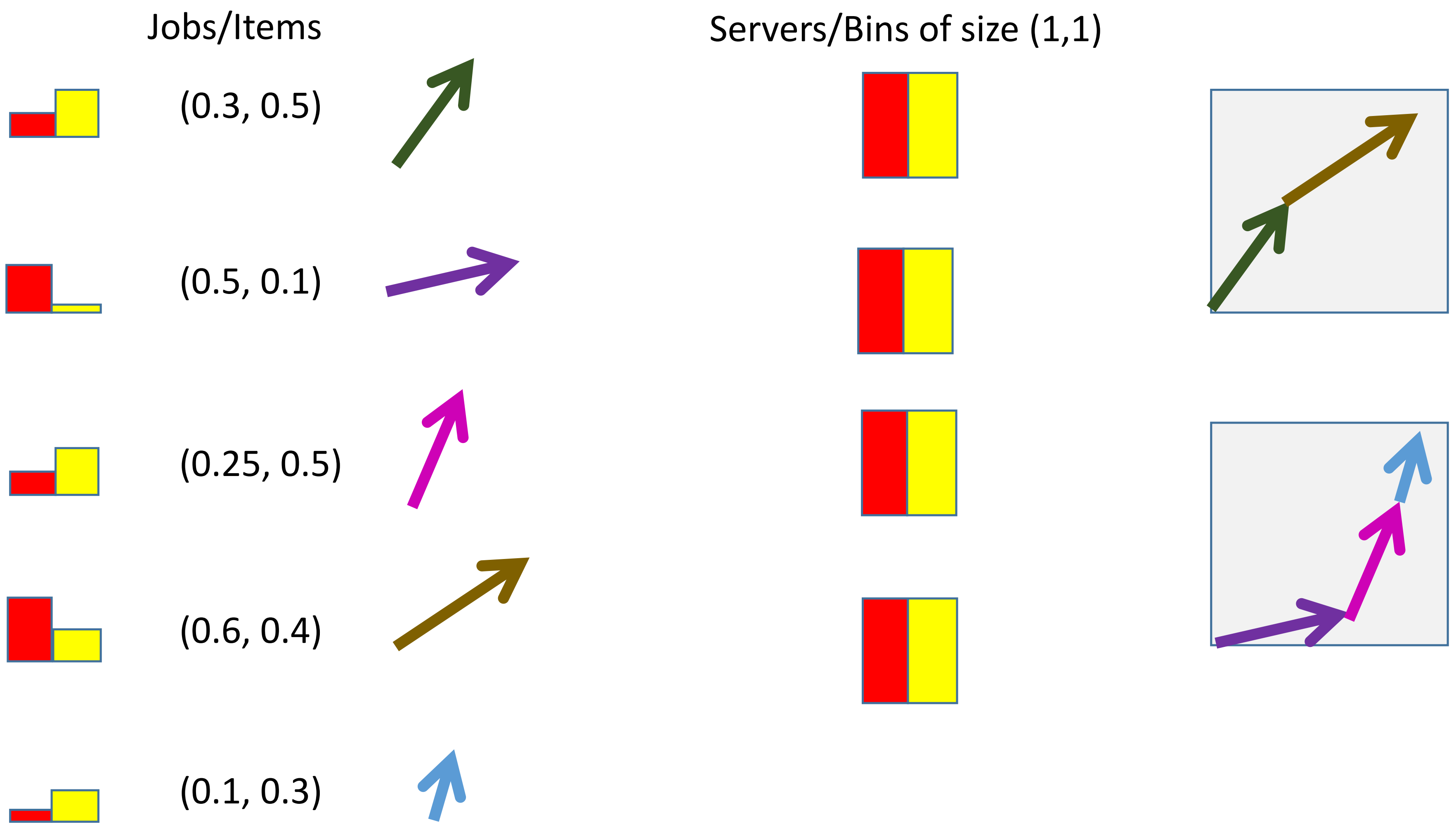
Vector packing: (Two Dimension)



Vector packing: (Two Dimension)

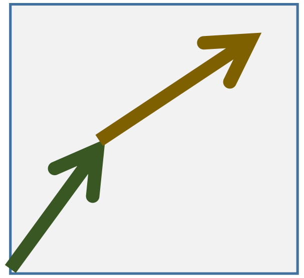


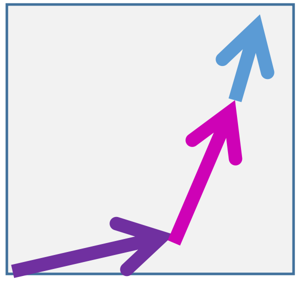
Vector packing: (Two Dimension)



Vector packing

Input:
Set of d-dimensional vectors with
nonnegative values.
Goal:
pack all vectors into minimum number
of unit vector bins such that for each bin for
each dimension for coordinate wise sum
of packed vector in it is ≤ 1.





Applications:

- Very Classical Generalization of Bin Packing.
  -- subsumes all applications of Bin Packing.
- Scheduling
  -- cloud computing.
- Vehicle Loading
- Layout Design

A tale of approximibility (asymptotic)

- Algorithm:
- $d + 0.7$ [FirstFit: Garey-Graham-Johnson-Yao 1976]
- $d + \epsilon$ [Linear grouping: Fernandez de la Vega-Lueker 1981]
- $2 + \ln(d) + \epsilon$ [Assignment LP: Chekuri-Khanna 1999]
- 2 for $d = 2$ [Kellerer-Kotov 2003]
- $1 + \ln(d) + \epsilon$ [Configuration LP: Bansal-Caprara-Sviridenko FOCS 2006]

- Hardness:
- No APTAS (from 3D Matching)[Woeginger 1997], Hardness=1.0001

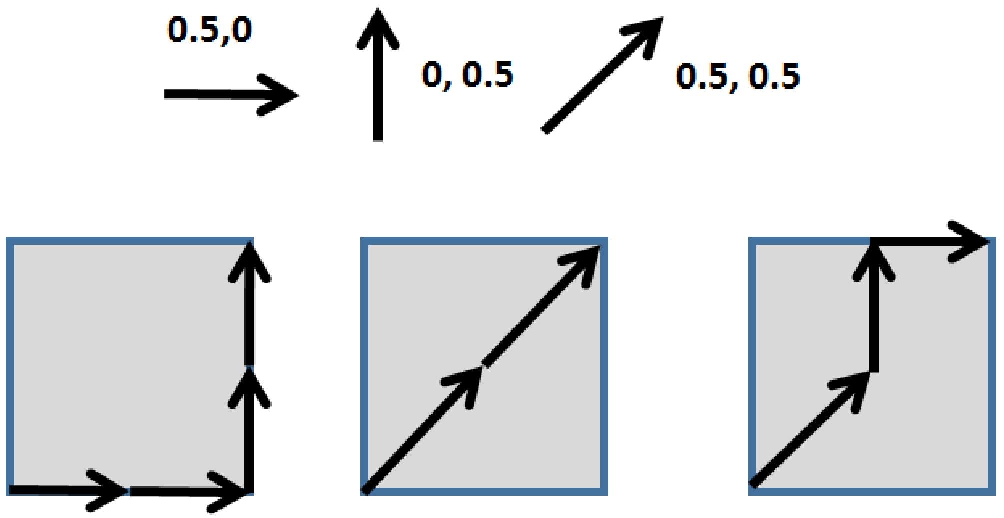
Our Results: [Bansal,Elias, K.]

- Algorithm:
- 1.405 Approximation Algorithm for 2 Dimensional Vector Packing.
- $1 + \ln\left(\frac{d+1}{2}\right)$ for d Dimensional Vector Packing.
- Hardness of d for constant rounding based algorithms.

- Resource Augmentation:
- If we allow extra resource of ϵ in $(d-1)$ dimensions, we can find a packing in polynomial time in $(1+\epsilon)Opt + O(1)$ number of bins.

Configurations

- $\mathbb{C}$*: set of configurations (possible way of feasibly packing a bin).*



Objective: min # configurations(bins)

Constraint:

For each item, at least one configuration containing the item should be selected.

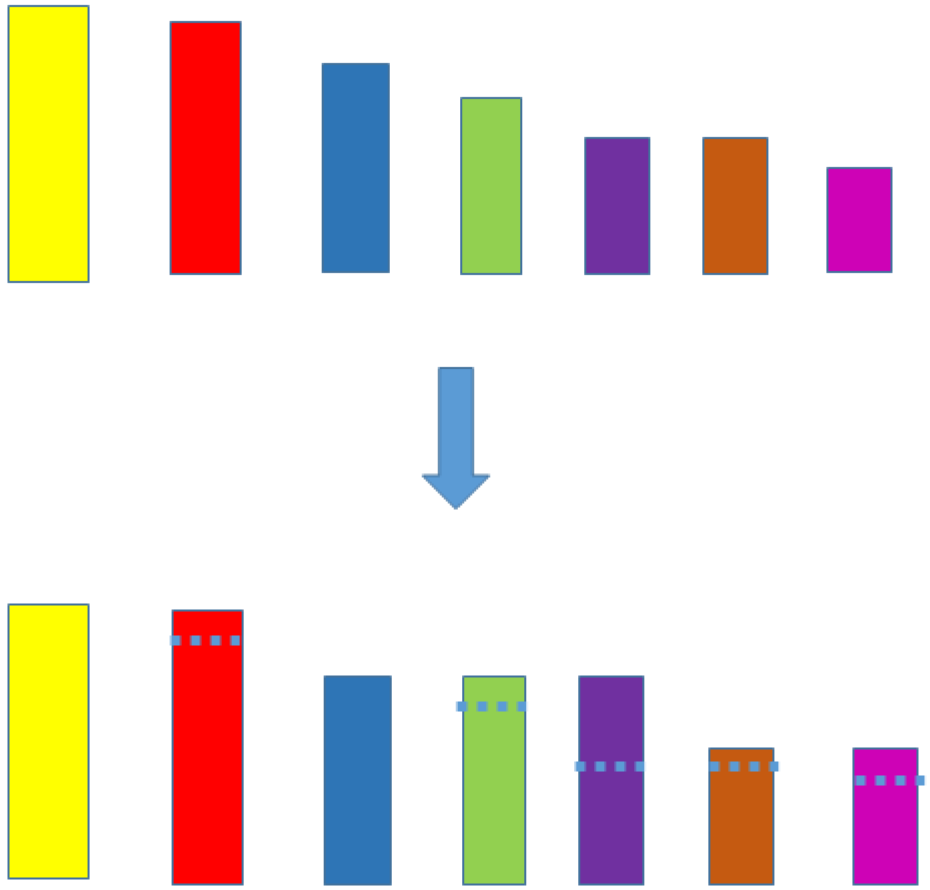
Constant Type of large items ⇒ polytime

- Assume all items are $\geq \epsilon$ in one of the dimensions, there are only constant $M = \frac{d}{\epsilon}$ items in each bin.
- If only T types of distinct items are there, possible number of configurations $\mathrm{R} = \binom{M+T}{M}$ is constant.
- The number of bins used is at most n
- Number of feasible packing is at most $= \binom{n+R}{R}$.
- Enumerating them and picking the best packing gives a polynomial time optimal algorithm.

O(1) Rounding based Algorithms

- Rounding up:
- Replace an item by a larger item.
- Loss:
  Due to larger items.
- Gain:

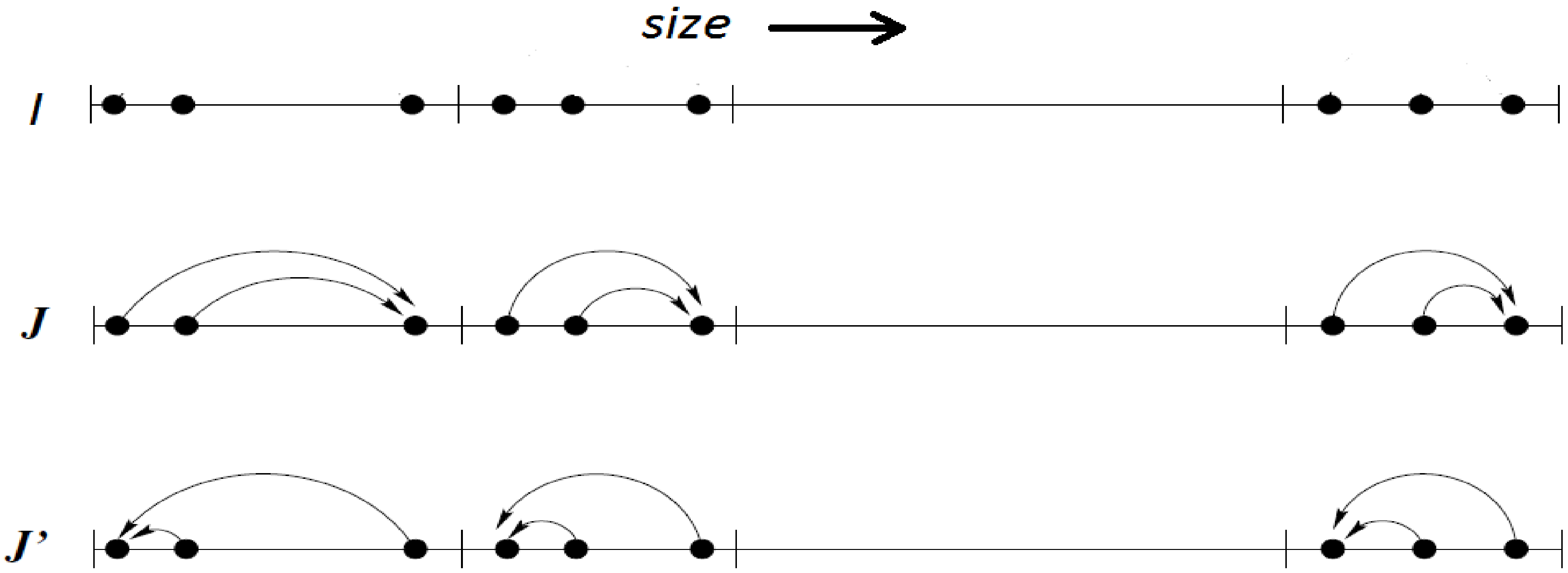
  Fewer configurations. If there are constant types of items we can solve rounded instance optimally.



Linear grouping (1D: If all items are $\geq \epsilon$) [Fernandez de la Vega –Lueker 1978]

Divide into $K = 1/\epsilon^2$ groups.

Each group contains at most $Q = n\,\epsilon^2$ items

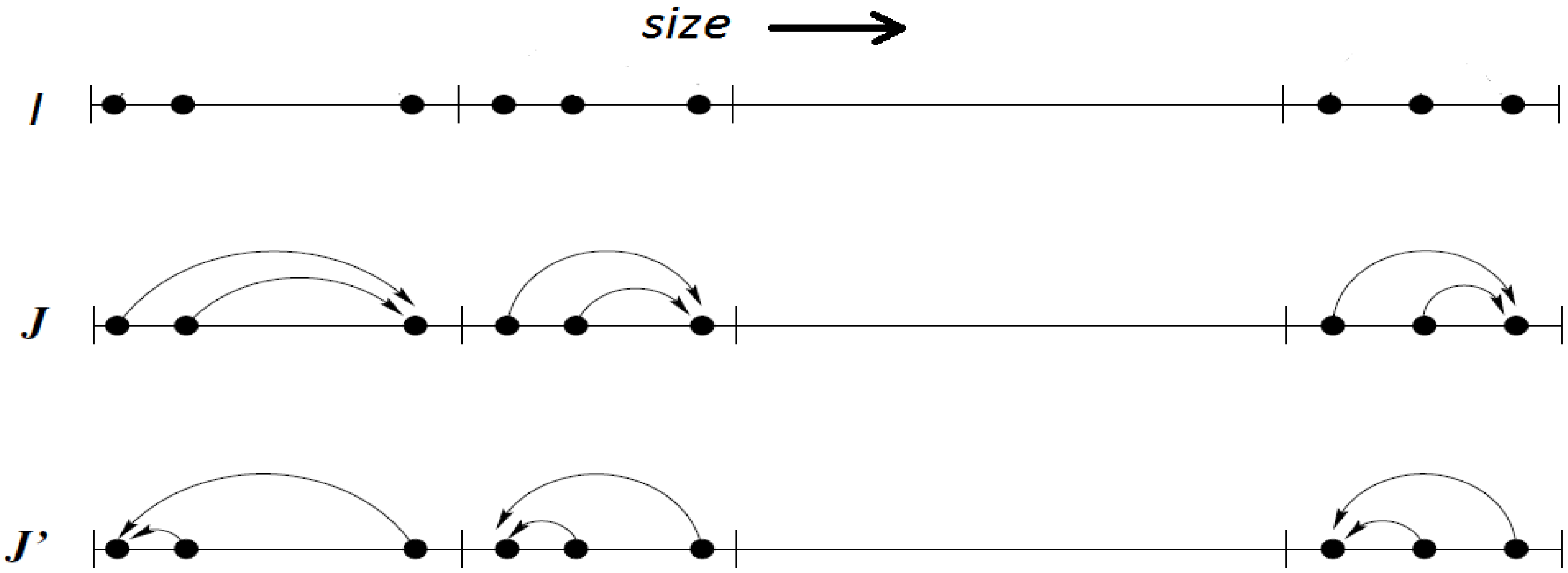


$$Opt(J') \leq Opt(I) \leq Opt(J) \leq Opt(J')(1+\epsilon)$$

Linear grouping (1D: If all items are $\geq \epsilon$) [Fernandez de la Vega –Lueker 1978]

Divide into $K = 1/\epsilon^2$ groups.

Each group contains at most $Q = n\,\epsilon^2$ items

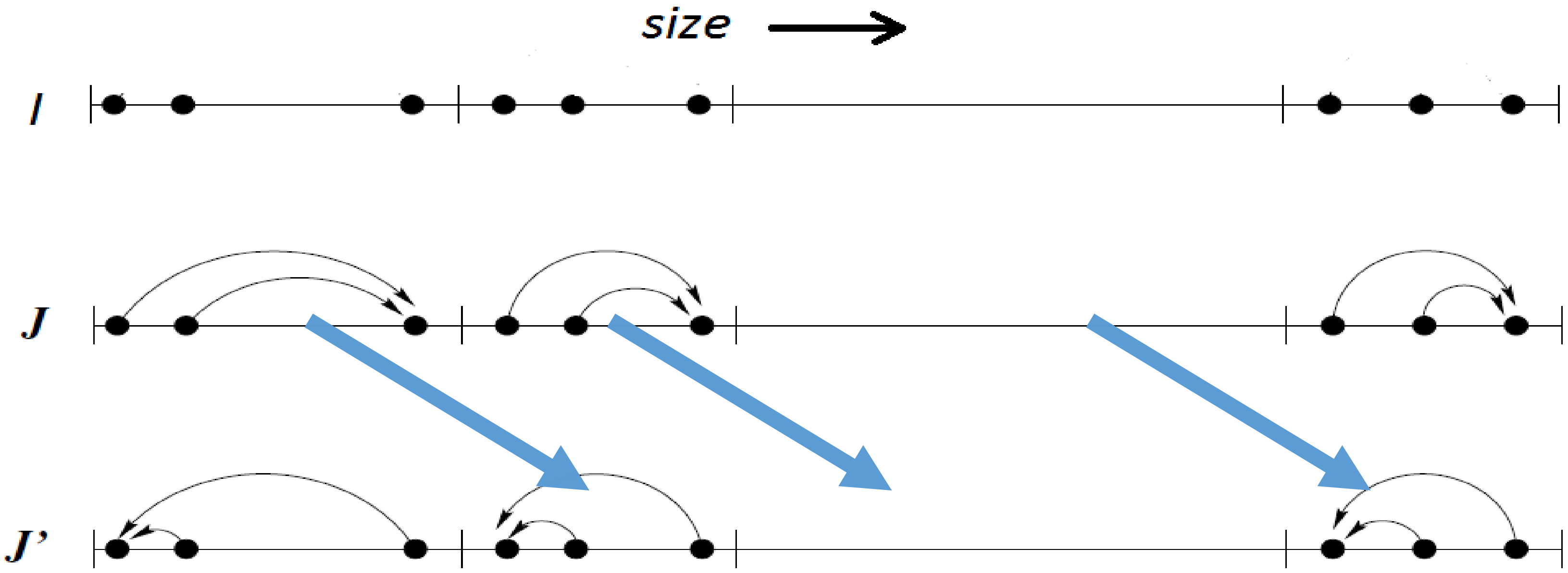


$$Opt(J') \leq Opt(I) \leq Opt(J)$$

Linear grouping (1D: If all items are $\geq \epsilon$) [Fernandez de la Vega –Lueker 1978]

Divide into $K = 1/\epsilon^2$ groups.

Each group contains at most $Q = n\,\epsilon^2$ items

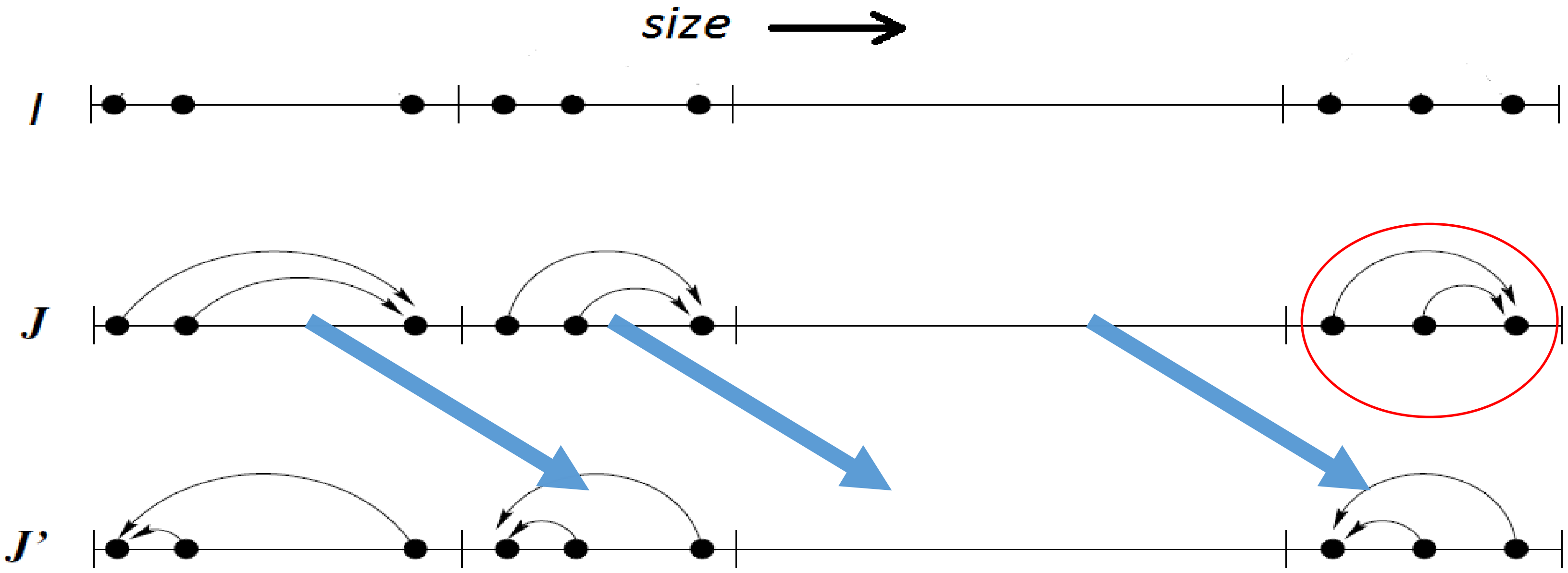


$$Opt(J') \leq Opt(I) \leq Opt(J)$$

Linear grouping (1D: If all items are $\geq \epsilon$) [Fernandez de la Vega –Lueker 1978]

Divide into $K = 1/\epsilon^2$ groups.

Each group contains at most $Q = n\,\epsilon^2$ items



$$Opt(J') \leq Opt(I) \leq Opt(J) \leq Opt(J') + Q$$

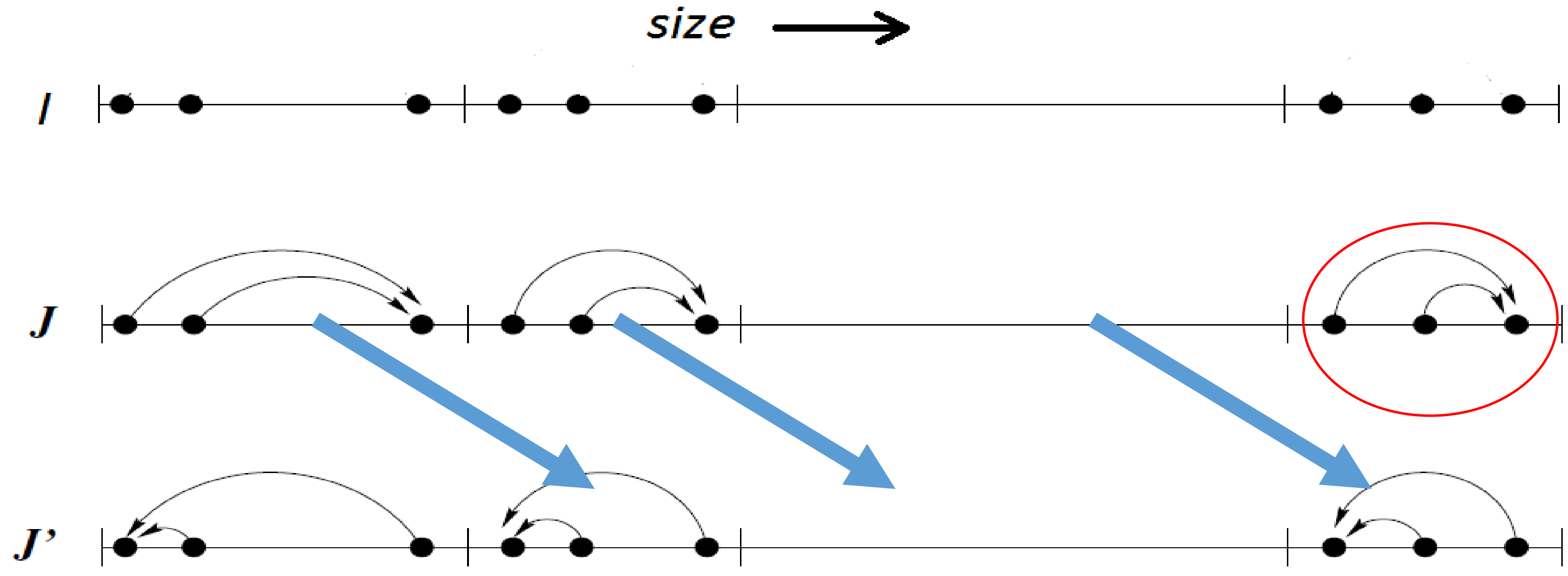
Linear grouping (1D: If all items are $\geq \epsilon$) [Fernandez de la Vega –Lueker 1978]

Large items:

$Opt \geq n\,\epsilon$

Extra items:

$Q = n\epsilon^2$
$\leq \epsilon Opt$



$$Opt(J') \leq Opt(I) \leq Opt(J) \leq Opt(J') + Q$$

Linear grouping (1D: If all items are $\geq \epsilon$) [Fernandez de la Vega –Lueker 1978]

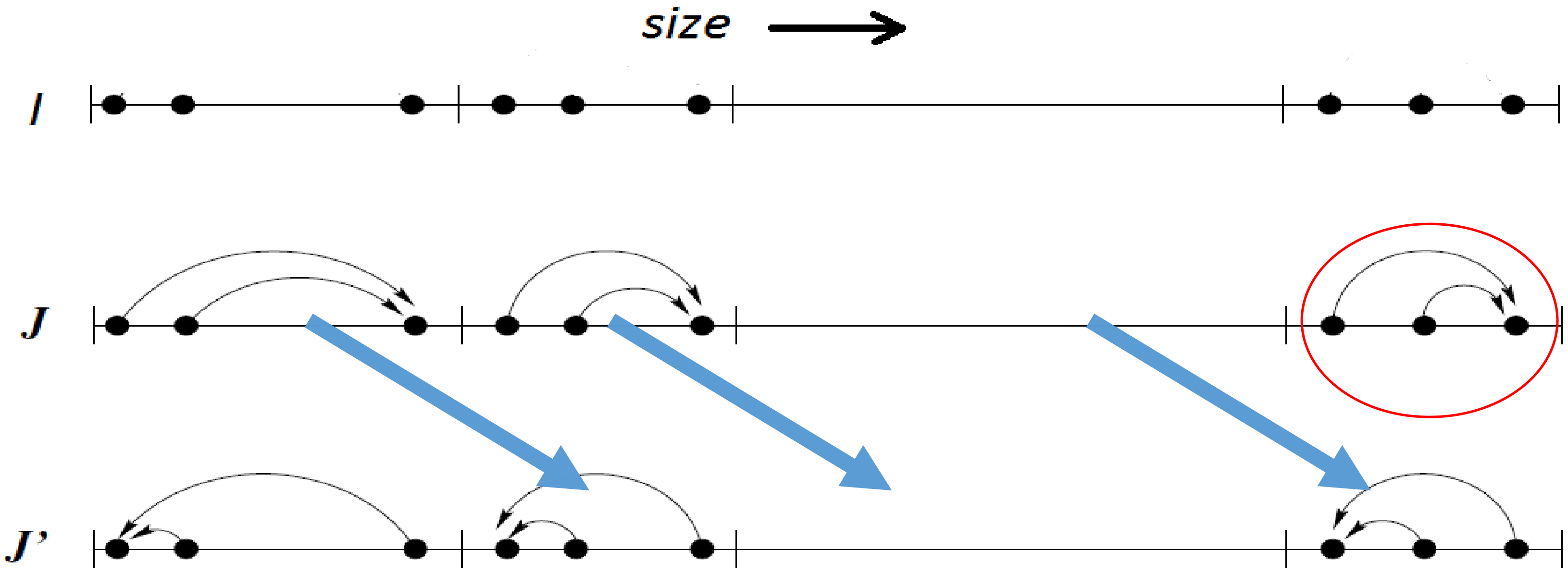
Large items:

$Opt \geq n\,\epsilon$

Extra items:

$Q = n\epsilon^2$

$\leq \epsilon Opt$



$$Opt(J') \leq Opt(I) \leq Opt(J) \leq Opt(J') + \epsilon Opt(J') \leq (1+\epsilon)Opt(J')$$

Configuration LP

- $\mathbb{C}$*: set of configurations(possible way of feasibly packing a bin).*

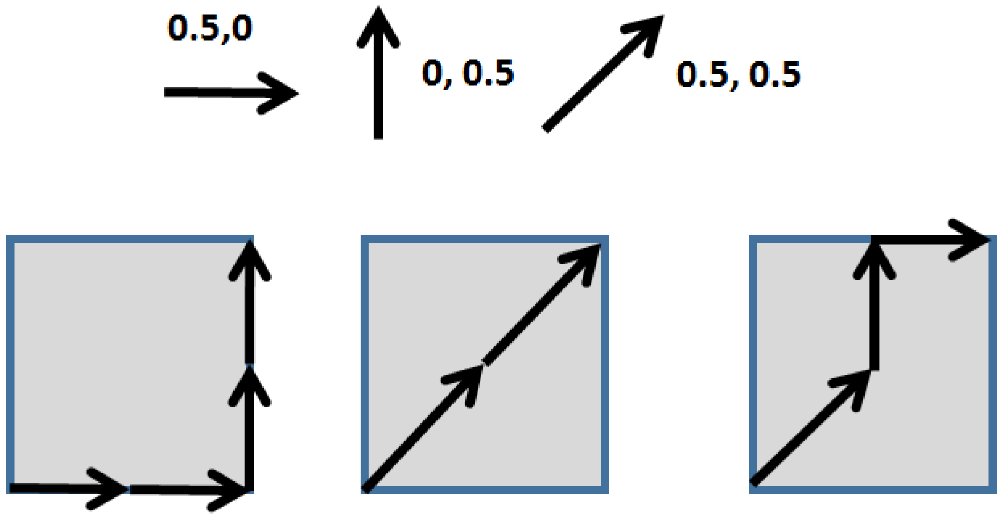
Primal:

$$\min \{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \ (i \in I), x_C \geq 0 \ (C \in \mathbb{C}) \}$$

Objective: min # configurations(bins)

Constraint:

For each item, at least one configuration containing the item should be selected.



Configuration LP

- $\mathbb{C}$: *set of configurations(possible way of feasibly packing a bin).*

Primal:

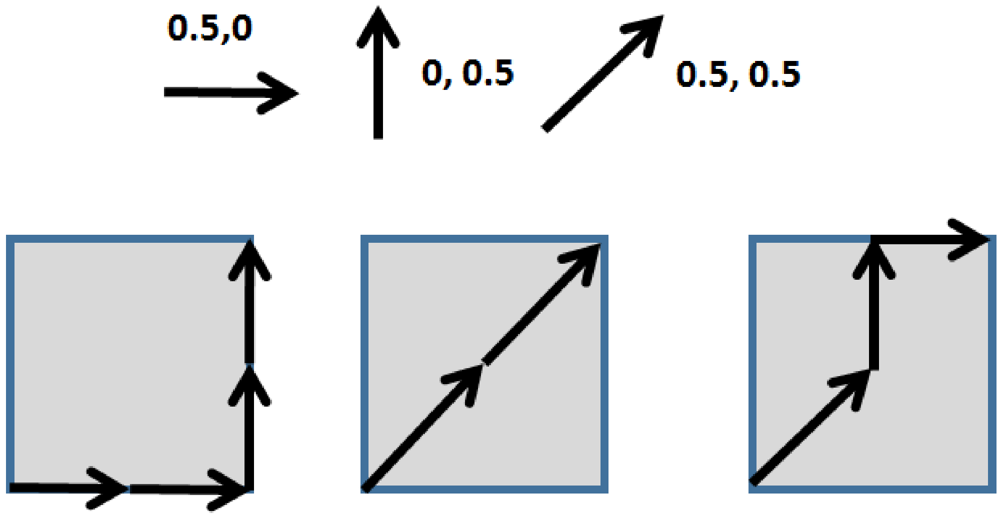
$$\min \left\{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \; (i \in I), x_C \geq 0 \; (C \in \mathbb{C}) \right\}$$

Gilmore Gomory LP:

Min $\{1^T x : \; Ax \geq b, \; x_C \geq 0 (C \in \mathbb{C})\}$

Columns: Feasible configurations

Rows: Items (or types of items)



Configuration LP

- $\mathbb{C}$*: set of configurations(possible way of feasibly packing a bin).*

Primal:

$$\min \{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \ (i \in I), x_C \geq 0 \ (C \in \mathbb{C}) \ \}$$

Dual:

$$\max \{ \sum_{i \in I} v_i : \sum_{i \in \mathrm{C}} v_i \leq 1 \ (C \in \mathbb{C}), v_i \geq 0 (i \in I) \}$$

- Problem: Exponential number of configurations!
- Solution: Can be solved within $(1+\epsilon)$ accuracy using separation problem for the dual.

Configuration LP

- $\mathbb{C}$*: set of configurations(possible way of feasibly packing a bin).*

Primal:

$$\min \{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \ (i \in I), x_C \geq 0 \ (C \in \mathbb{C}) \}$$

Dual:

$$\max \{ \sum_{i \in I} v_i : \sum_{i \in C} v_i \leq 1 \ (C \in \mathbb{C}), v_i \geq 0 (i \in I) \}$$

Dual Separation problem =>
d-D Vector Knapsack problem:

$$\sum_{i \in C} v_i^* > 1 \ (for\ some\ C \in \mathbb{C})$$

- Problem: Exponential number of configurations!
- Solution: Can be solved within $(1+\epsilon)$ accuracy using separation problem for the dual.

Configuration LP

- $\mathbb{C}$*: set of configurations(possible way of feasibly packing a bin).*

Primal:

$$\min \{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \ (i \in I), x_C \geq 0 \ (C \in \mathbb{C}) \}$$

Dual:

$$\max \{ \sum_{i \in I} v_i : \sum_{i \in C} v_i \leq 1 \ (C \in \mathbb{C}), v_i \geq 0 (i \in I) \}$$

Dual Separation problem =>
d-D Vector Knapsack problem:

$$\sum_{i \in C} v_i^* > 1 \ (for\ some\ C \in \mathbb{C})$$

- Max $\sum_{\{i \in I\}} v_i^* x_i$
- S.t. $\sum_{\{i \in I\}} s_i^k x_i \leq 1$, for $k \in [d]$.

$$x_i \in \{0,1\}$$

- Problem: Exponential number of configurations!
- Solution: Can be solved within $(1 + \epsilon)$ accuracy using separation problem for the dual.

Randomized Rounding of Configuration LP

- 1. Solve configuration LP using APTAS. Let $z^* = \sum_{\{C \in \mathbb{C}\}} x_C^*$.

Primal:

$$\min \left\{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \; (\, i \in I), x_C \geq 0 \; (C \in \mathbb{C}) \right\}$$

Randomized Rounding of Configuration LP

- 1. Solve configuration LP using APTAS. Let $z^* = \sum_{\{C \in \mathbb{C}\}} x_C^*$.
- 2. Round: For $\lceil \ln \rho \, . \, z^* \rceil$ iterations :

  select a configuration C' at random with probability $\frac{x_{C'}^*}{z^*}$.

Primal:

$$\min \{ \sum_{C} x_C : \sum_{C \ni i} x_C \geq 1 \; (i \in I), x_C \geq 0 \; (C \in \mathbb{C}) \}$$

Round and Approx Framework (R & A) [Bansal-Caprara-Sviridenko06, Bansal-K. 14]

- 1. Solve configuration LP using APTAS. Let $z^* = \sum_{\{C \in \mathbb{C}\}} x_C^*$.
- 2. Round: For $\lceil \ln \rho \,.\, z^* \rceil$ iterations :

  select a configuration C' at random with probability $\frac{x_{C'}^*}{z^*}$.
- 3. Let S be the set of remaining uncovered elements.

  Pack them using a O(1) rounding based algorithm.

Few Residual Items!

- $\mathbb{P}(i \in S) = \left(1 - \sum_{\{C \ni i\}} \frac{x_C^*}{z^*}\right)^{\lceil (\ln \rho) z^* \rceil}$
- $\leq e^{-\ln \rho}$
- $= \frac{1}{\rho}$.

- Opt still might not shrink!

R & A for constant rounding based Algorithms

- $\tilde{I}$: instance obtained from I by rounding up big items. t is constant.

- Configuration LP($\tilde{I}$):
- $Minimize\ \sum_r x_r$
- $\sum_r c_{B_j}^r x_r \geq |B_j| \; \forall\, j \in [t]$
- $x_r \geq 0\ (r = 0,1\ \ \ldots m)$

- Configuration LP($\tilde{I} \cap S$):
- $Minimize\ \sum_r x_r$
- $\sum_r c_{B_j}^r x_r \geq |B_j \cap S| \; \forall\, j \in [t]$
- $x_r \geq 0\ (r = 0,1\ \ \ldots m)$

Fact: Any feasible system $(b \in \mathbb{R}^n), Ax = b, x \geq 0$
has a solution x* with support(x*) $\leq n$.

Proof Sketch

- Rounding based Algo : O(1)types of items $= O(1)$ number of constraints in Configuration LP.
- $ALGO(S) \approx OPT(\tilde{S}) \approx LP(\tilde{S})$.
- As # items for each item type shrinks by ρ, $LP(\tilde{S}) \approx \frac{1+\epsilon}{\rho} LP(\tilde{I})$.
- ρ Approximation: $LP(\tilde{I}) \leq \rho\, OPT(I) + O(1)$.

- $ALGO(S) \approx OPT(\tilde{S}) \approx OPT(I)$.

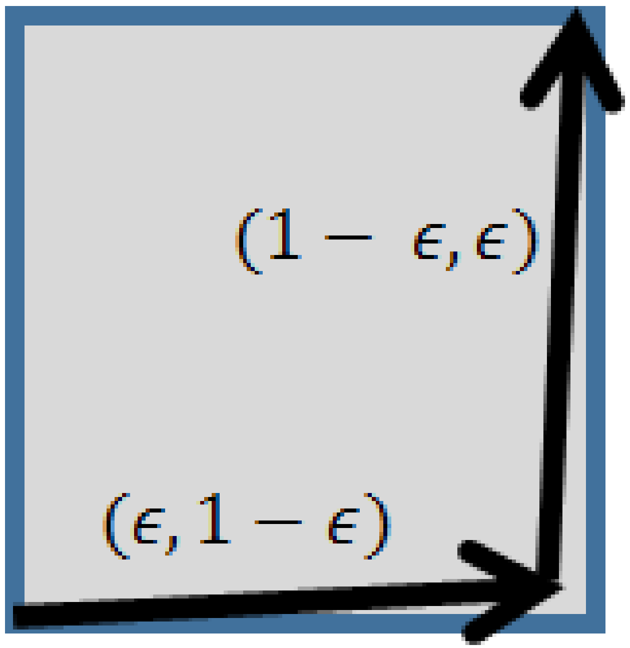
Proof Sketch

- Thm: R&A gives a $(1 + \ln \rho)$ approximation.
- Proof:
- Randomized Rounding : q= $\ln \rho . LP(I)$
- Residual Instance S = $(1 + \epsilon)OPT(I) + O(1)$.

- Round + Approx => $(\ln \rho + 1 + \epsilon)OPT(I) + O(1)$.

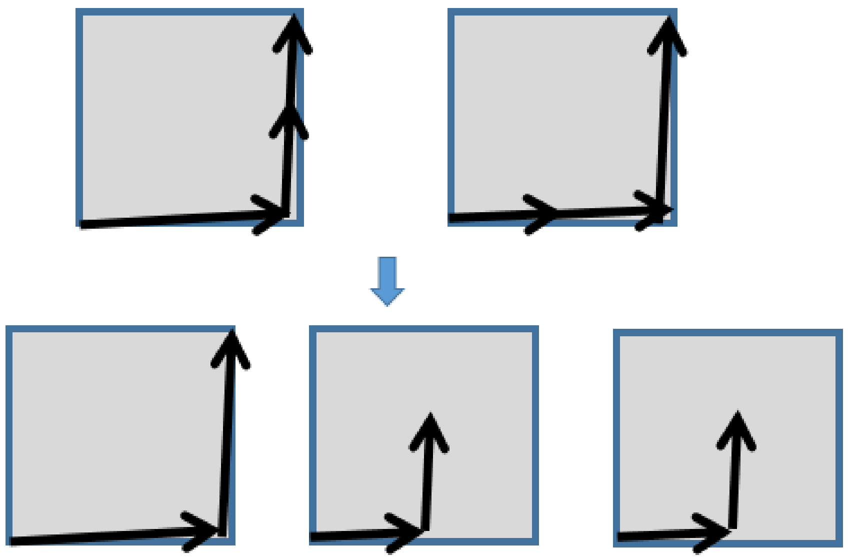
Round and Approx Framework

- Theorem: $(d+\epsilon)$ approximation is tight for $O(1)$ rounding based algorithms.
- $(1+\ln d+\epsilon)$ approx. is tight in this framework.
- $d=2$
- For, 2 dimensions matching bins (one or two items cover $1-\epsilon$ area of bins) create problem.
- If there are such bins we can get a 3/2 approximation using a structural lemma.



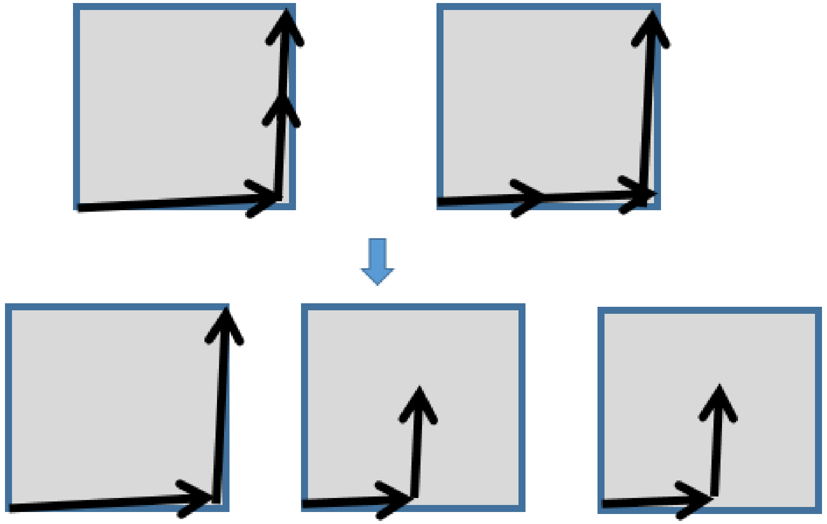
Structural lemma

- 2 D vector packing:
- Any packing of m bins can be transformed into a packing of $3m(1+\epsilon)/2$ bins where each bin either contains 2 items (matching bins) or has slack in (d-1) dimensions (nonmatching bins).
- d D vector packing:
- Any packing of m bins can be transformed into a packing of packing of $2m\left(1+\frac{\epsilon}{2}\right)$ bins where at most m bins contain $\leq d-1$ items (compact bins) in it and other $m(1+\epsilon)$ bins have slack in $(d-1)$ dimensions (noncompact bins).



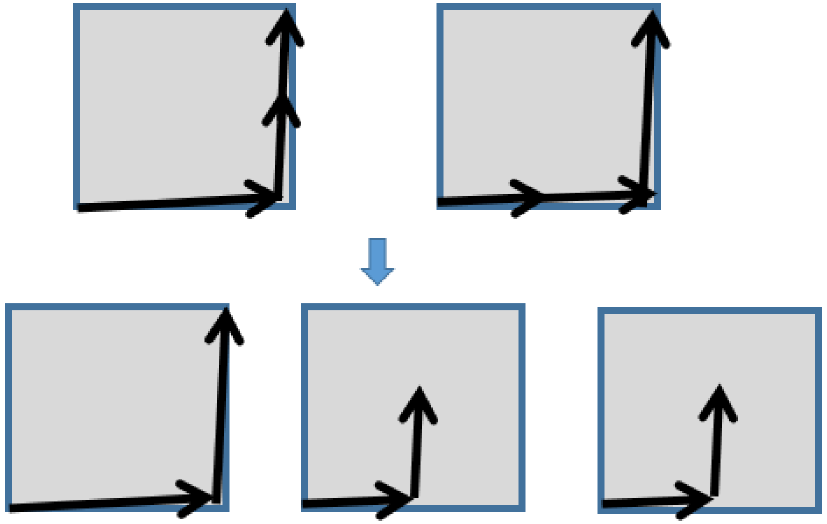
Proof Sketch

- 2 D vector packing: 2 bins A,B => 3 bins D,E,F s.t. each bin either has 2 items or has slack in (d-1) dim.
- $\geq$ 3 items in a bin A/B.
- In each dim at most one item $\geq \frac{1}{2}$.
- There is one item $\leq \left(\frac{1}{2}, \frac{1}{2}\right)$. [candidate item c_A, c_B]
- Two candidate items c_A, c_B can be packed in one matching bin D.
- Big Items: ($> \epsilon$ in one of the dimensions)
- Small items: ($< \epsilon$ in all dimensions)



Proof Sketch

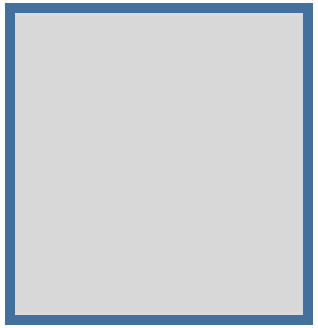
- 2 D vector packing:
- one item $\leq \left(\frac{1}{2}, \frac{1}{2}\right)$ in each bin [candidate item]
- Two candidate items c_A, c_B can be packed in one matching bin D.
- Big Items: ($> \epsilon$ in one of the dimensions)
- Small items: ($< \epsilon$ in all dimensions)
- If candidate item big, we get enough slack.
- Otherwise, we can remove a subset of small items from A (and B) to get bins with slack E,F and pack removed items in D.



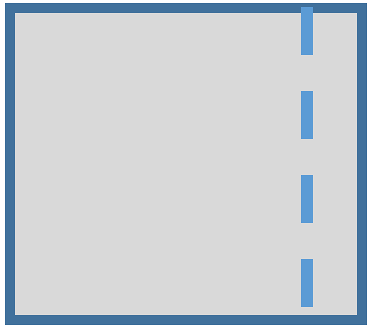
Resource Augmentation

- Allow extra resource ϵ in one dimension.

- Theorem:
  If we allow resource augmentation in (d-1) dimensions we can pack items in $(1+\epsilon)Opt$ number of bins.



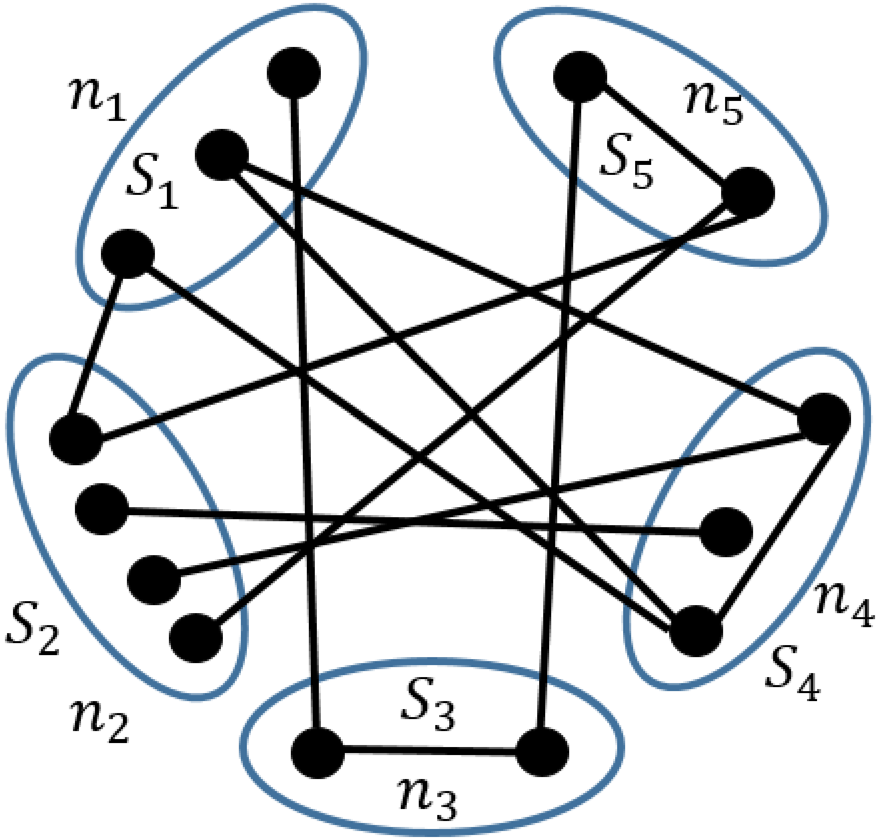
- Round big items Linear grouping in non-augmented dimension, Round down items to multiple of ϵ^2 in other dimensions ⇒ O(1) types of big items.
- Find optimal packing of rounded big items.
- Pack small items using an assignment LP.



- 2 D vector packing: (Existential Result)
- Any packing of m bins can be transformed into a packing of $3m(1+\epsilon)/2$ bins where each bin either contains 2 items (matching bins) or has O(1) types of big items (nonmatching bins).
- We can find these rounded values in polynomial time.
- If we can separate out items in matching bins and nonmatching bins, we can pack each of them separately.
- Not possible! So we guess number of items in matching bins and nonmatching bins for each rounded class.

MultiObjective MultiBudget Matching [Chekuri-Vondrak-Zenklusen]

- Given a graph and a partition of its vertices into k sets such that $V := S_1 \cup S_2 \cup \cdots S_k$ and numbers $n_1, n_2, \cdots n_k$; there is a polynomial time Algorithm that finds a matching that saturates at least $(1-\epsilon)n_i$ and at most n_i items from each S_i.
- If no such solution is found, the algorithm returns a certificate that there is no such feasible matching for the instance.



11

Algorithm for 2D Vector Packing

- 1. Guess Opt, number of $matching$ bins M and $nonmatching$ bins N.
- 2. Find round vectors r^x and assign items to the corresponding class W^x .
- 3. Find m^x, n^x: the number of items from matching and nonmatching bins in W^x.
- 4. Use multi objective multi budget matching to pack m^x items from each class W^x.
- 5. Solve configuration LP restricted to the remaining items. Let $z^* = \sum_{\{C \in \mathbb{C}\}} x^*_C$.
- 6. Round: For $\lceil \ln \rho \,.\, z^* \rceil$ iterations : (Take $\rho = \frac{3}{2}$)
  select a configuration C' at random with probability $\frac{x^*_{C'}}{z^*}$.
- 7. Approx: Let S be the set of remaining uncovered elements.
  Apply $3/2$ approximation algorithm A on S that rounds the big items to $O(1)$ types and small items are packed near-optimally using an assignment LP.

- Theorem: Above algorithm always returns a solution with at most $(1 + \ln\left(\frac{3}{2}\right) + \epsilon)Opt$ bins.
- M+$\ln \rho$. (M+N)+ N

Algorithm for d Dimensional Vector Packing

- 1. Guess Opt, number of bins in the optimal solution.
- 2. Find round vectors r^x and assign items to the corresponding class W^x .
- 3. Solve configuration LP. Let $z^* = \sum_{\{C \in \mathbb{C}\}} x_C^*$.
- 4. Round: For $\lceil \ln \rho \, . \, z^* \rceil$ iterations (Take $\rho = \frac{d}{2}$) :

  select a configuration C' at random with probability $\frac{x^*_{C'}}{z^*}$.
- 5. Let S be the set of remaining uncovered elements. Find m^x, n^x: the number of items from compact and noncompact bins in $(W^x \cap S)$.
- 6. Use multi objective multi budget matching to pack m^x items from each class W^x into $1.5\ Opt$ bins.
- 7. Apply $O(1)$ rounding based approximation algorithm A on S that rounds the big items to $O(1)$ types and

  pack remaining big items in $\frac{2(1+\epsilon)Opt}{d}$ bins. Pack small items near-optimally using an assignment LP.

- Theorem: Above algorithm always returns a solution with at most $\left(1.5 + \ln\left(\frac{d}{2}\right) + o_d(1)\right) Opt$ bins.

Related Open problems

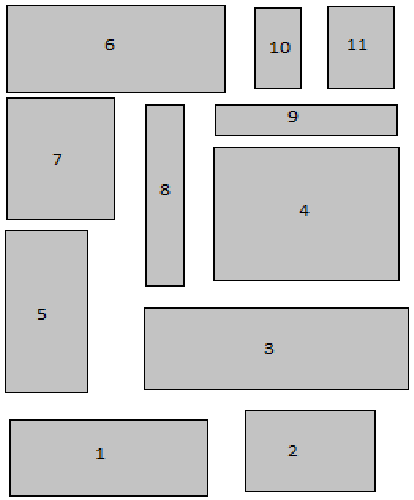
Vector Bin Covering and Maxmin Scheduling

- Vector Bin Covering: Partition vectors such that in each set in each dimension sum of vectors is $\geq\ 1$.
- Random partition into $2 \ln d$ sets work with high probability! (balls/bins)
- Can we get $lnlnd$?
- PTAS for Multidimensional minimum knapsack?
- Generalizations to unrelated/related job scheduling.

Two-Dimensional Geometric Bin Packing

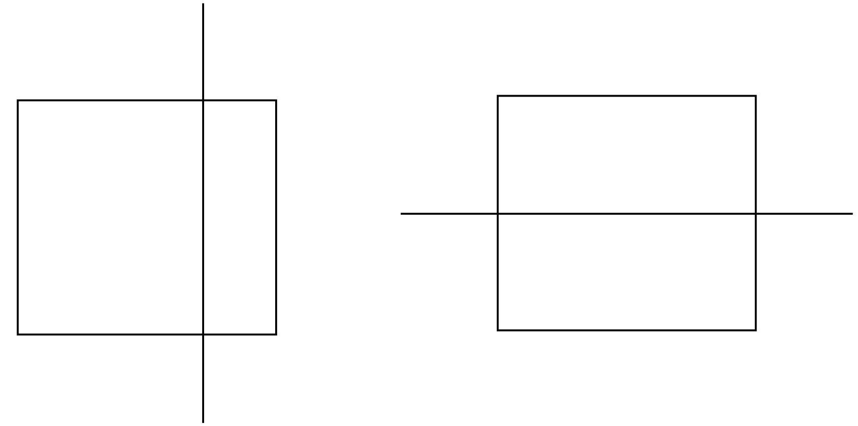
- Given: Collection of rectangles (by width, height)
- Goal: Pack them into minimum number of unit square bins.

- Orthogonal Packing: rectangles packed parallel to bin edges.
- With 90 degree *Rotations* and *without rotations*.



Guillotine Packing

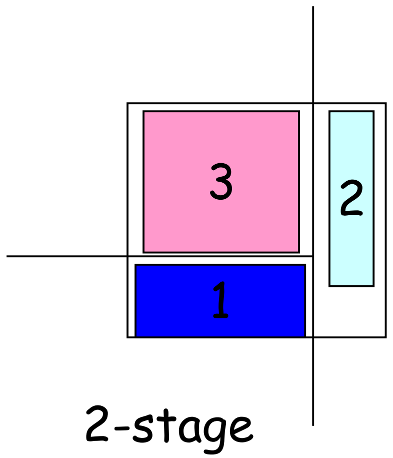
Guillotine Cut: Edge to Edge cut across a bin

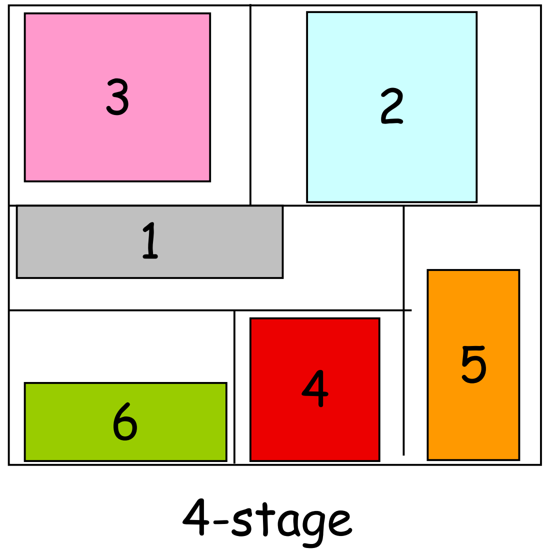


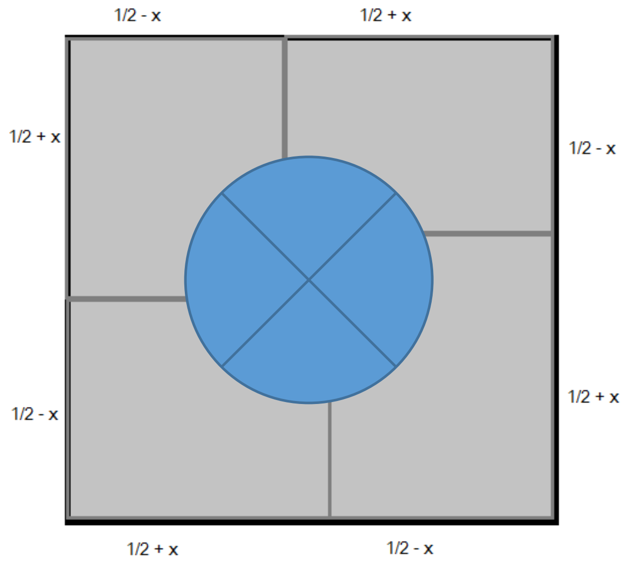
Objective: Minimize number of bins such that packing in each bin is a guillotine packing.

Guillotine Packing => General Bin packing

Guillotine Cut: Edge to Edge cut across a bin





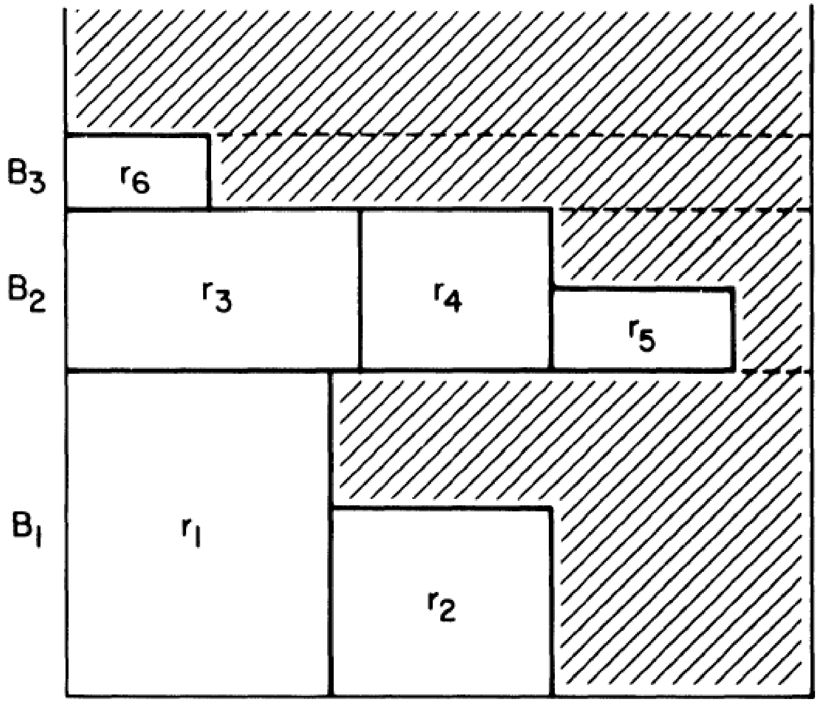


- There is an APTAS for Guillotine Packing [BLS FOCS 2005].
- Given any packing of m bins, there is a Guillotine packing in 4m/3 bins. => 4/3 approximation.
- PTAS for geometric 2 D knapsack

Questions!

Thank You!

Next Fit Decreasing Height(NFDH)



- Considered items in a non-increasing order of height and greedily packs items into shelves.
- Shelf is a row of items having their bases on a line that is either the base of the bin or the line drawn at the top of the highest item packed in the shelf below.
- items are packed left-justified starting from bottom-left corner of the bin, until the next item does not fit. Then the shelf is closed and the next item is used to define a new shelf whose base touches the tallest(left most) item of the previous shelf.
- If the shelf does not fit into the bin, the bin is closed and a new bin is opened. The procedure continues till all the items are packed.

- If we pack small rectangles ($w, h \leq \delta$) using NFDH into B, total $w.h - (w+h).\delta$ area can be packed.

Details of Chernoff Bound

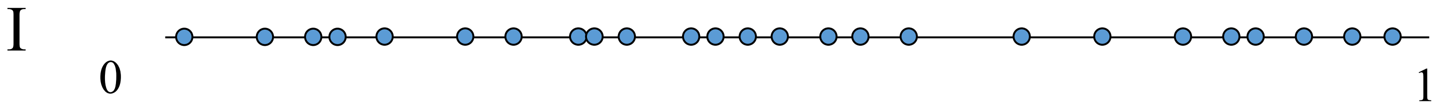
- $|B\_j|, w(L\_k)$ and $h(W_\ell)$ are at least $\Omega\left(\left(\frac{1}{\epsilon^2}\right)\log t\right)$

- By standard Chernoff bounds, $\mathbb{P}\ [B_j \cap\ S\Big|\geq\ (1+\epsilon)|B_j \cap\ S|\}]$ is at most \exp$(-\epsilon^2\ |B_j|/\rho)$ = \exp$(-\Omega(\log t)/\rho)\ =\ 1/poly(t)$.

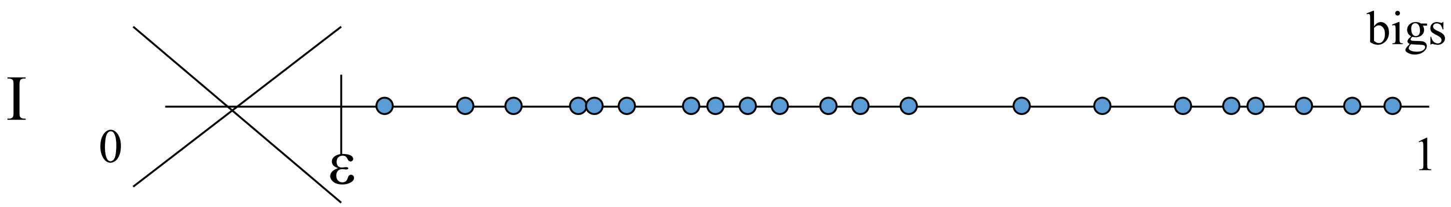
- Taking a union bound over the t constraints, whp, RHS for each constraint in LP($\tilde{I} \cap S$) is at most $(1+\epsilon)/\rho$ times the right hand side of the corresponding constraint in LP($\tilde{I}$).

- 1 D BP: FDLVGL, linear grouping. Karp Karmarkar.
- Partition Hardness.
- 2D history
- Config lp 1 slide
- RandA 2 slides
- Proof outline 4slides
- 3/2 algo idea 2 slides
- 4/3 hardness 2 slides
- 3/2 hardness 1 slides

1-d: Algorithm

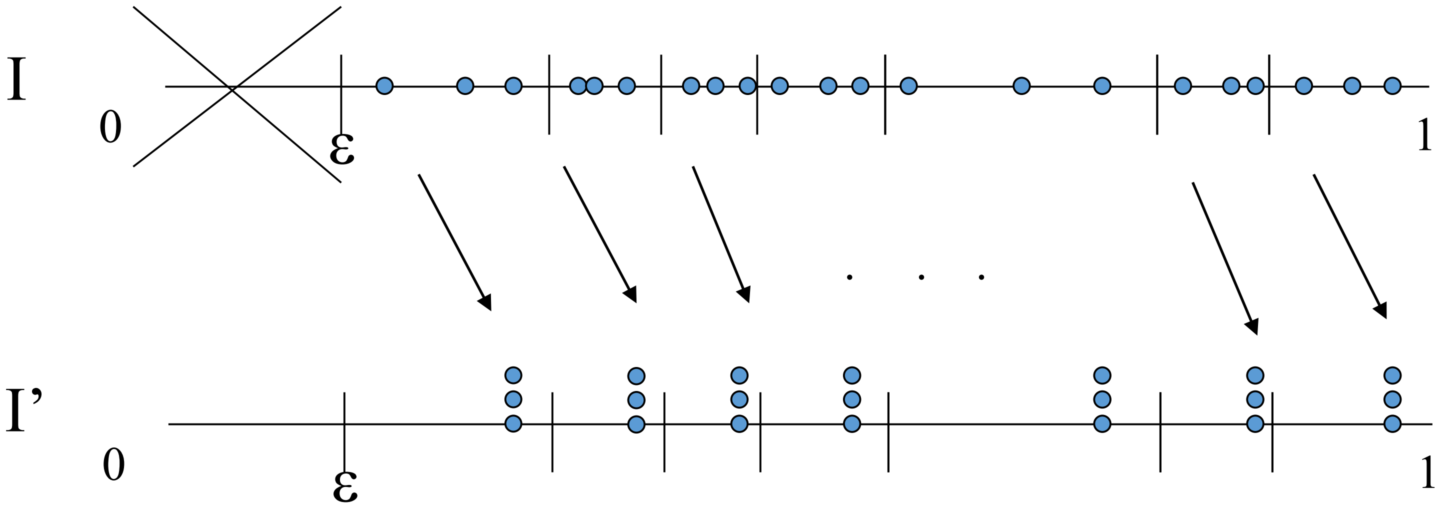


1-d: Algorithm



1-d: Algorithm

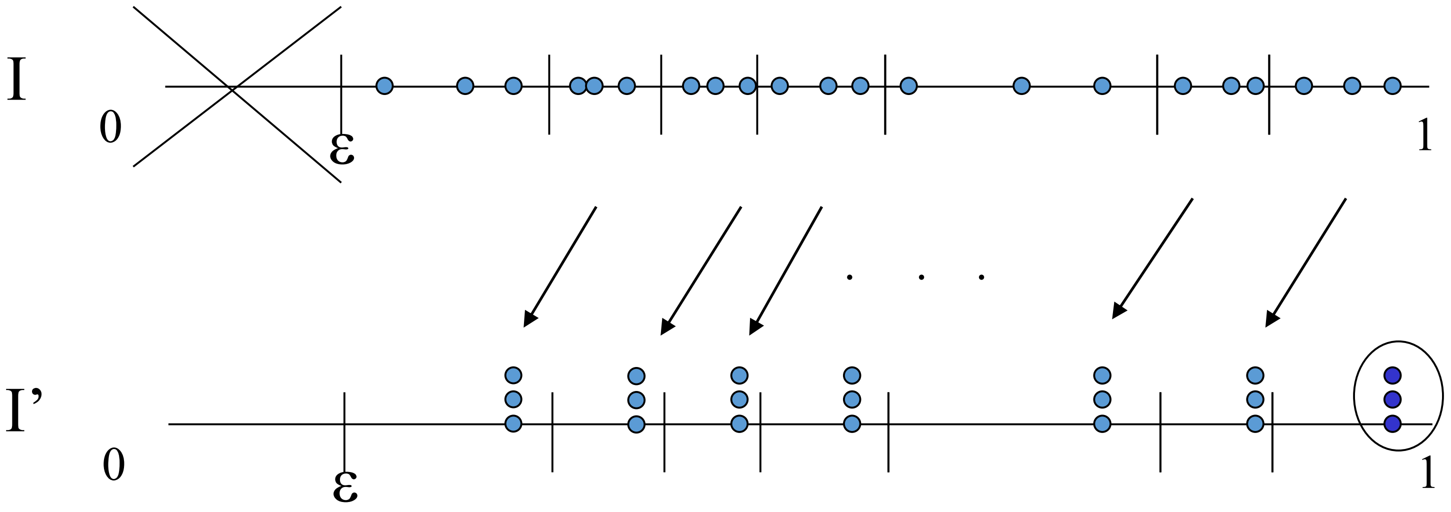
Partition bigs into $1/\varepsilon^2$ = O(1) groups, with equal objects



I’ ¸ I

1-d: Algorithm

Partition bigs into $1/\varepsilon^2$ = O(1) groups, with equal objects



I’ ¸ I

I’ – { } · I

I’ ¼ I

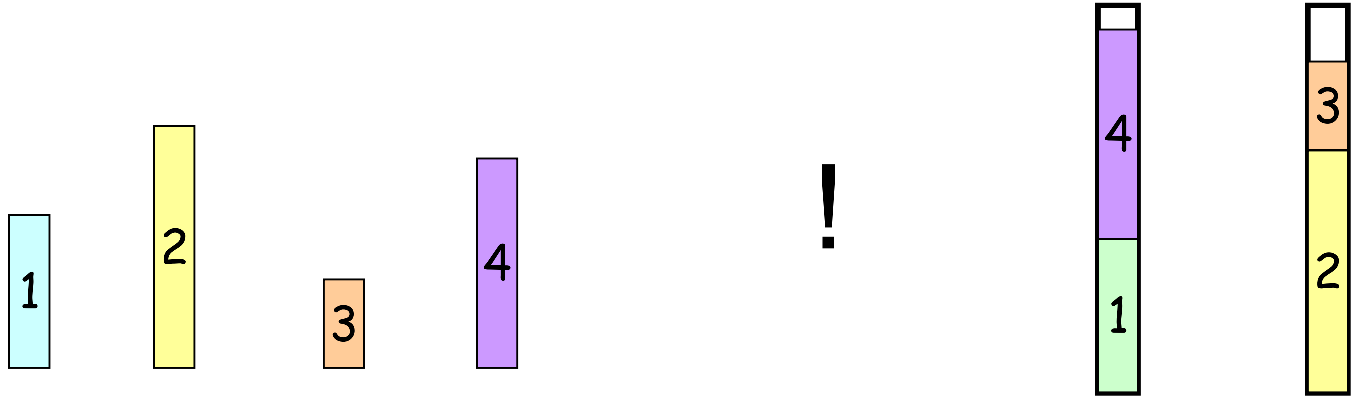
I’ has only $O(1/\varepsilon^2)$ distinct sizes

APTAS for 1-d bin packing

Theorem: [de la Vega, Lueker 81]

$$\mathrm{Alg}(I) \cdot \ \mathrm{Opt}(I)/(1-\varepsilon) + 1/\varepsilon^2$$

$$¼ \ \mathrm{Opt}(I)\,(1+\varepsilon) + f(\varepsilon)$$



Main idea

Simplify Original instance I -> I’

- I’: easy to solve
- Solutions of I and I’ close (within $1+\varepsilon$)

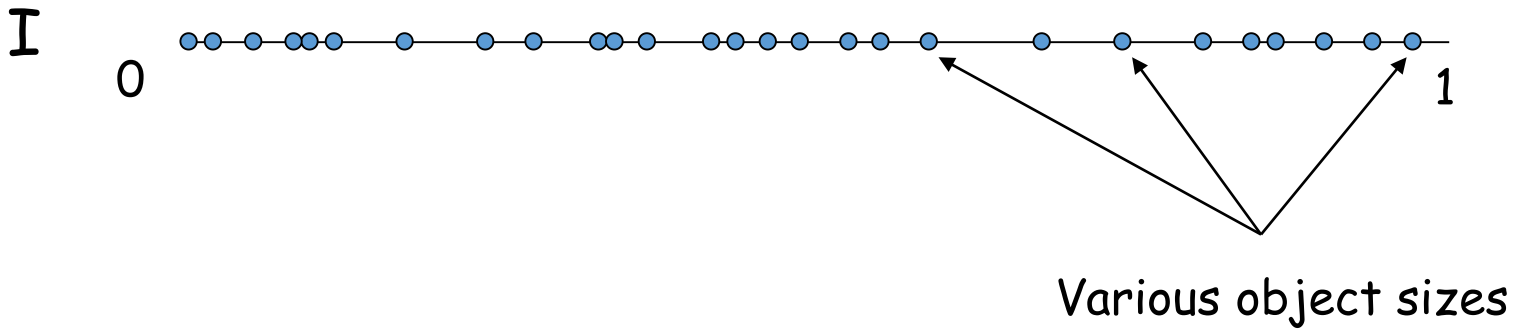
Ideas applied to 1-d packing

· ε: Small ¸ ε: Big

1) I ! I' with 1/ε^2 different big sizes & solns. within 1+ε

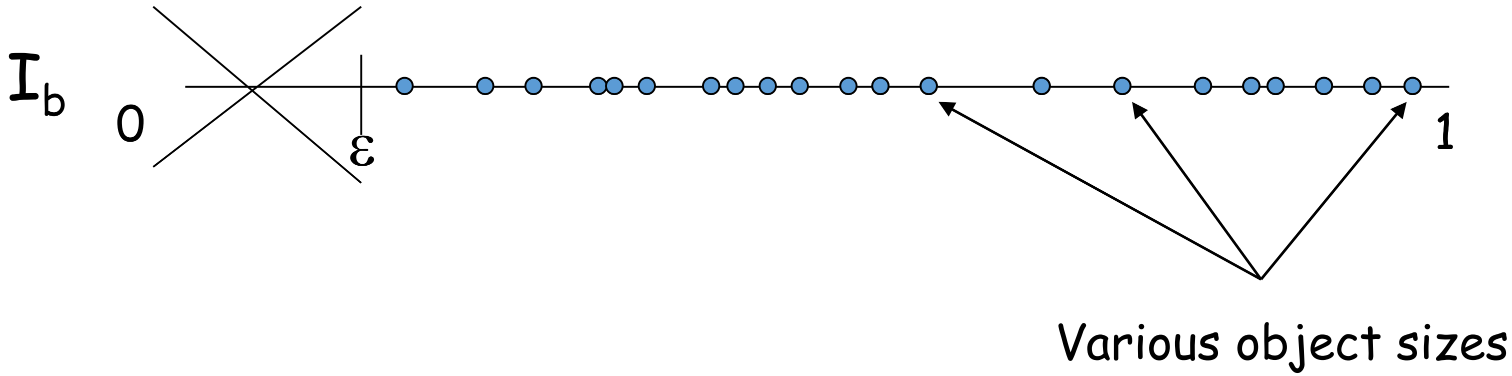
2) I' easy: If k= O(1) different big sizes, can get Opt + k

1-d: Rounding to a simpler instance



1-d: Rounding to a simpler instance

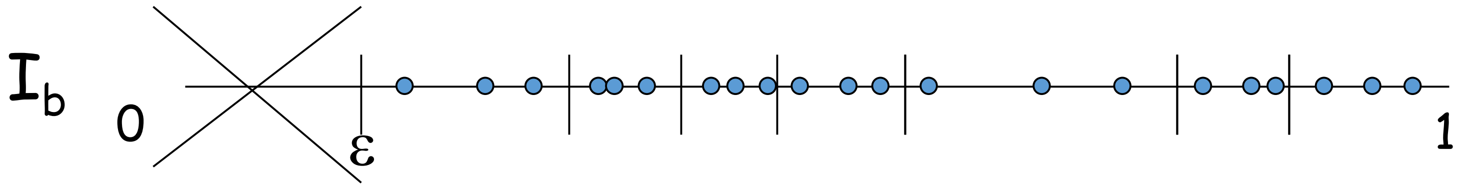
I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)



1-d: Rounding to a simpler instance

I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)

Partition big into $1/\varepsilon^2$ groups, each group has b¢ ε^2 objects

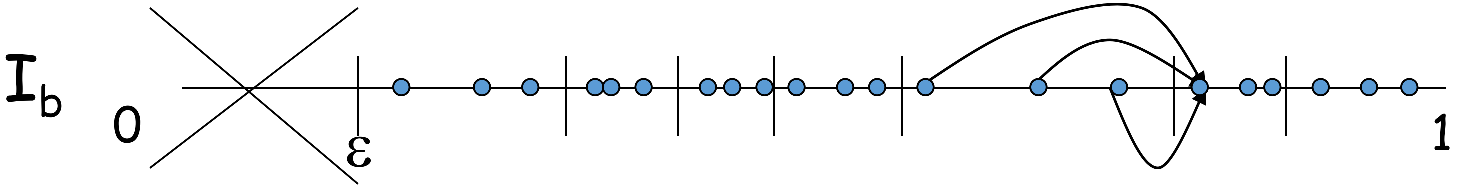


(here b¢ε^2 =3)

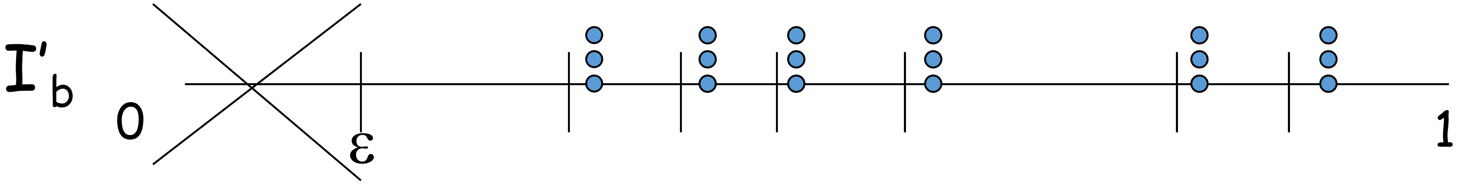
1-d: Rounding to a simpler instance

I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)

Partition big into $1/\varepsilon^2$ groups, each group has $b ¢ \varepsilon^2$ objects



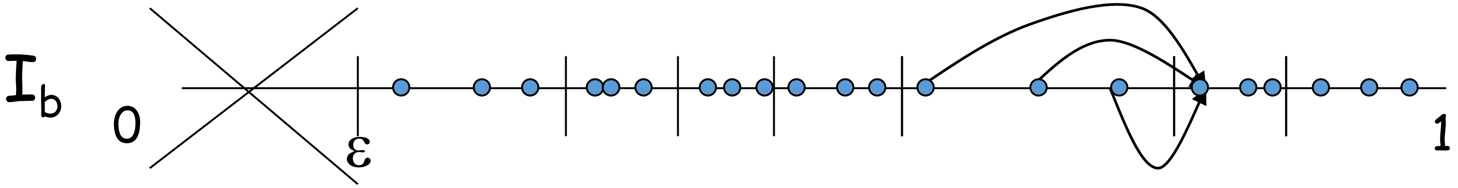
Instance I'_b : Ignore largest $b ¢ \varepsilon^2$ objects.
Round up sizes to smallest size in next higher group



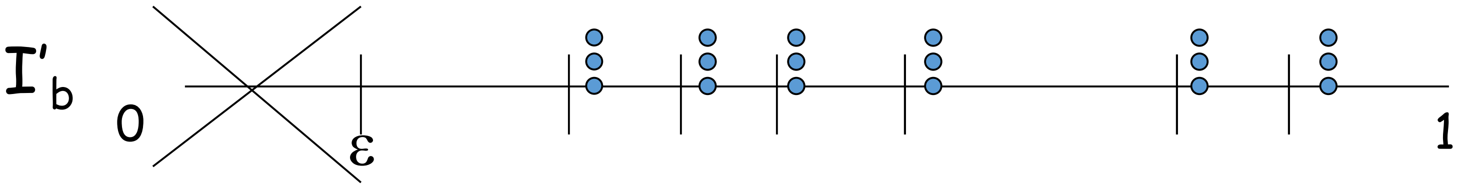
1-d: Rounding to a simpler instance

I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)

Partition big into $1/\varepsilon^2$ groups, each group has b¢ ε^2 objects



Instance I'_b : Ignore largest b ¢ ε^2 objects.
Round up sizes to smallest size in next higher group

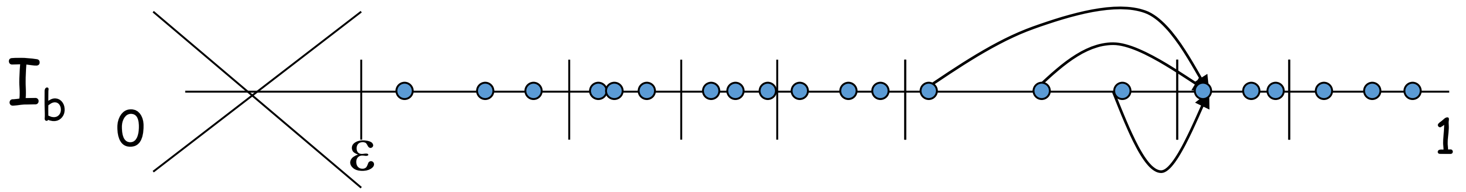


Alg(I'_b) · Alg(I_b)

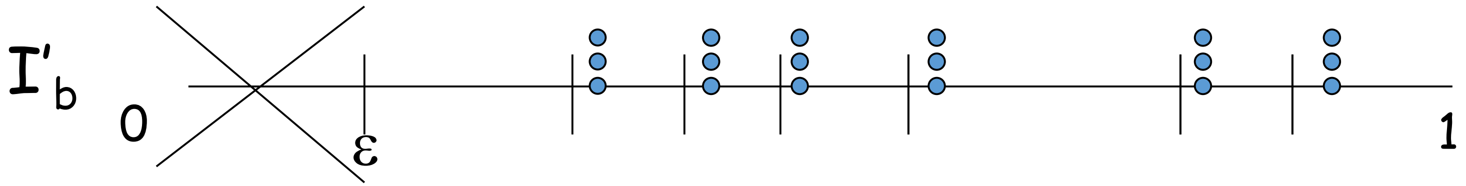
1-d: Rounding to a simpler instance

I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)

Partition big into $1/\varepsilon^2$ groups, each group has b¢ ε^2 objects



Instance I'_b : Ignore largest b ¢ ε^2 objects.
Round up sizes to smallest size in next higher group

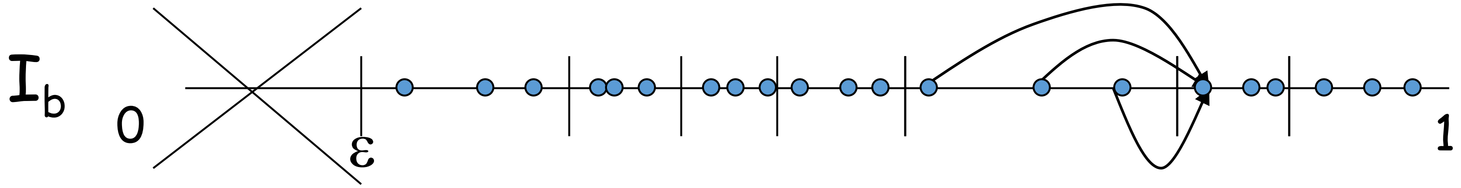


Alg(I_b) · Alg(I'_b) + b ε^2

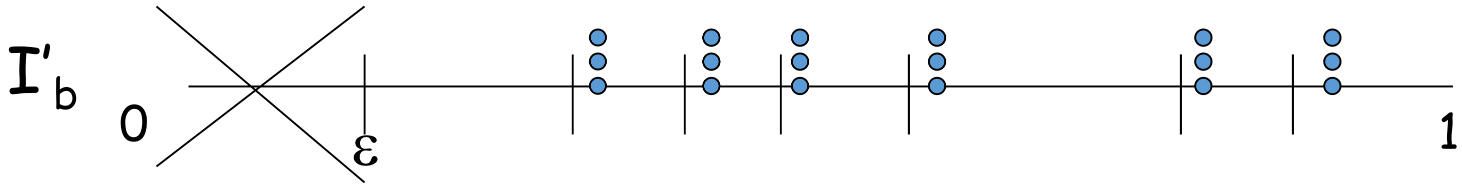
1-d: Rounding to a simpler instance

I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)

Partition big into $1/\varepsilon^2$ groups, each group has $b¢\, \varepsilon^2$ objects



Instance I'_b : Ignore largest $b ¢\, \varepsilon^2$ objects.
Round up sizes to smallest size in next higher group

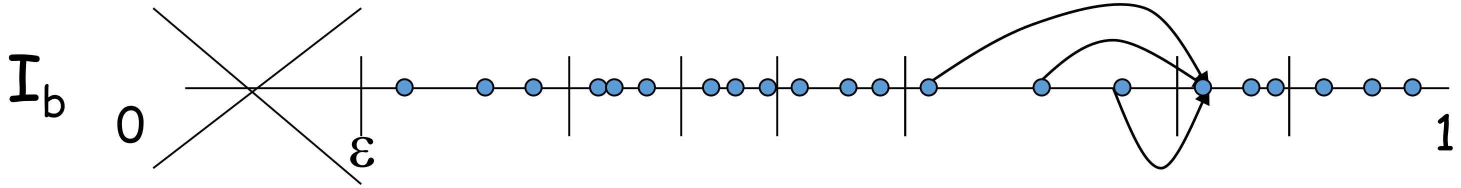


$Alg(I'_b) \cdot Alg(I_b) \cdot Alg(I'_b) + b\, \varepsilon^2$

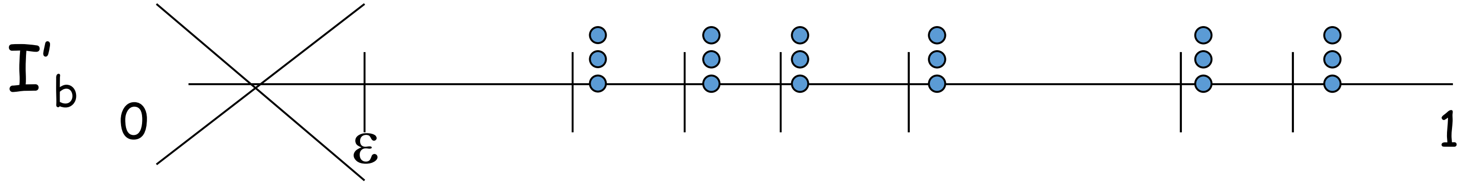
1-d: Rounding to a simpler instance

I_b : I restricted to bigs. Let b = # of bigs (i.e. ¸ ε)

Partition big into $1/\varepsilon^2$ groups, each group has b¢ ε^2 objects



Instance I'_b : Ignore largest b ¢ ε^2 objects.
Round up sizes to smallest size in next higher group



Alg(I'_b) · Alg(I_b) · Alg(I'_b) + b ε^2

Opt(I_b) ¸ b ε) b ε^2 · ε Opt(I_b)

1-d: Solving the “few and big” case

I'_b : $1/\varepsilon^2$ different sizes > ε. Call these $s_1,\ldots,s_k$.

Configuration: A way to pack a bin (Eg: $C = [3\ s_1, 17\ s_3, 5\ s_{18}]$)

|Configurations| · $(1/\varepsilon^2)^{1/\varepsilon}$ = O(1)

x_i : # of bins with configuration i

n_j : # of objects of size s_j in instance

c_{ij}: # of objects of size s_j in configuration i.

Minimize $\sum_i x_i$

$\sum_i c_{ij} x_i$ ¸ n_j 8 j 2 $[1,..,1/\varepsilon^2]$

x_i ¸ 0   8 i, x_i 2 Z

IP for I'_b

1-d: “Few and Big” using LP

Minimize $\sum_i x_i$

$\sum_i c_{ij} x_i$ ¸ n_j 8 j 2 $[1,..,1/\varepsilon^2]$

x_i ¸ 0   (Relaxed to be fractional)

Clearly, LP (I_b’) · OPT(I_b’)

x_i could be fractional.

Round up to next integer (Eg: 17.34 -> 18)

Adds · # configurations = $(1/\varepsilon^2)^{1/\varepsilon}$ = O(1)

In fact, adds · $1/\varepsilon^2$ (non-zero x_i’s in basic soln)

1-d: Filling in the smalls

So, $\mathrm{Alg}(I_b) \cdot \mathrm{Opt}(I)/(1-\varepsilon) + 1/\varepsilon^2$

Packing smalls:

- In each bin, fill as many smalls as possible.
- If bins not enough, open new bins to fill smalls.

Proof:

- If no new bins opened, done.
- If new bins opened, all bins (except maybe last) filled $1-\varepsilon$

$$\text{So, } \mathrm{Alg}(I) \cdot \mathrm{Area}(I)/(1-\varepsilon) + 1$$
$$\cdot \mathrm{Opt}(I)/(1-\varepsilon) + 1$$

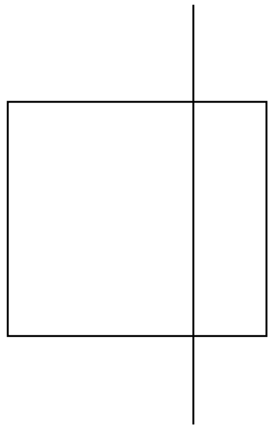
1-d: Overview

0) Partition into small and big
1) Pack small objects later
2) Round large objects to O(1) sizes.

   Solve the “few and big” case almost optimally.

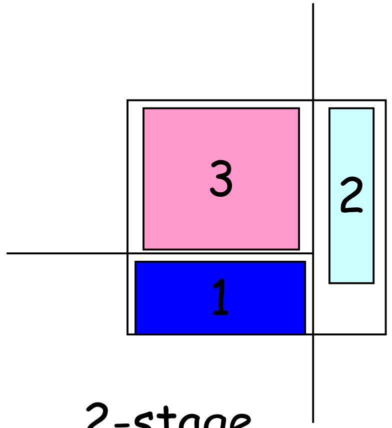
Guillotine Bin Packing

Guillotine Cut: Edge to Edge cut across a bin

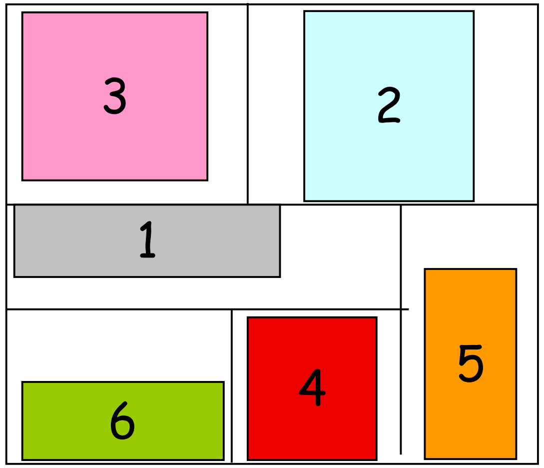


Guillotine Bin Packing

Guillotine Cut: Edge to Edge cut across a bin



2-stage



4-stage

k-stage Guillotine Packing [Gilmore, Gomory]

k recursive levels of guillotine cuts to recover all items.

Non-guillotine Packing