Lecture 3: Approximation Algorithms for Packing Problems in Two Dimensions

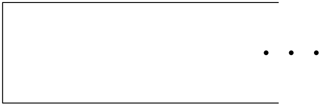
Helmut Alt, Freie Universität Berlin
at 10th Winter School on Computational Geometry
Amirkabir University of Technology, Tehran

2 March 2018

Strip Packing

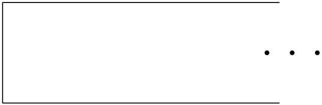
Strip Packing

Consider a strip S of fixed height.

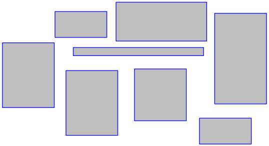


Strip Packing

Consider a strip S of fixed height.

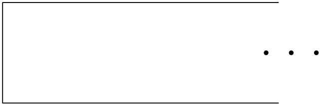


Given: A set $\mathcal{O}$ of axes-parallel rectangles.

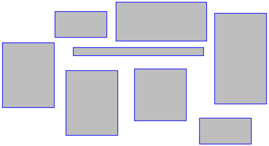


Strip Packing

Consider a strip S of fixed height.



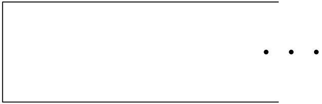
Given: A set $\mathcal{O}$ of axes-parallel rectangles.



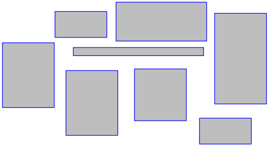
Determine: Packing of $\mathcal{O}$ into S of minimum length.

Strip Packing

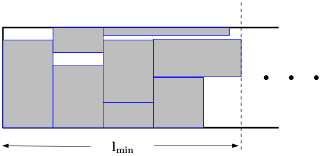
Consider a strip S of fixed height.



Given: A set $\mathcal{O}$ of axes-parallel rectangles.



Determine: Packing of $\mathcal{O}$ into S of minimum length.



Strip Packing Algorithms

Strip Packing Algorithms

many heuristics that yield constant factor approximations

Strip Packing Algorithms

many heuristics that yield constant factor approximations

simplest ones:

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
    1. Sort the rectangles of $\mathcal{O}$ non-increasing width.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.
  3. If no more space on top, start a new level right of the previous one and continue.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.
  3. If no more space on top, start a new level right of the previous one and continue.
- First-Fit.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.
  3. If no more space on top, start a new level right of the previous one and continue.
- First-Fit.

  Similar to Next-Fit.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.
  3. If no more space on top, start a new level right of the previous one and continue.
- First-Fit.
  Similar to Next-Fit.
  The next rectangle to be packed will be placed as low as possible in the leftmost level in which it fits.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.
  3. If no more space on top, start a new level right of the previous one and continue.
- First-Fit.
  Similar to Next-Fit.
  The next rectangle to be packed will be placed as low as possible in the leftmost level in which it fits. A new level is defined if no level has enough space left for the new rectangle.

Strip Packing Algorithms

many heuristics that yield constant factor approximations
simplest ones:

- Next-Fit.
  1. Sort the rectangles of $\mathcal{O}$ non-increasing width.
  2. Pack the rectangles at the left end of the strip from bottom to top.
  3. If no more space on top, start a new level right of the previous one and continue.
- First-Fit.
  Similar to Next-Fit.
  The next rectangle to be packed will be placed as low as possible in the leftmost level in which it fits. A new level is defined if no level has enough space left for the new rectangle.

both are approximation algorithms with factors 3 and 2.7 for FF and NF, respectively !

Reduction of Minimum Rectangle to Strip Packing

Reduction of Minimum Rectangle to Strip Packing

Facts:

Reduction of Minimum Rectangle to Strip Packing

Facts:

- Strip packing is NP-hard.

Reduction of Minimum Rectangle to Strip Packing

Facts:

- Strip packing is NP-hard.
- If there is an efficient α-approximation algorithm for strip packing then for any $\delta > 0$ there is an efficient $(1+\delta)\alpha$-approximation algorithm for finding the minimum area a.p. rectangle for packing a.p. rectangles under translation.

Reduction of Minimum Rectangle to Strip Packing

Facts:

- Strip packing is NP-hard.
- If there is an efficient α-approximation algorithm for strip packing then for any $\delta > 0$ there is an efficient $(1+\delta)\alpha$-approximation algorithm for finding the minimum area a.p. rectangle for packing a.p. rectangles under translation.

Algorithm:

Let h_0 be the minimum height of the rectangles, H the sum of all heights.

Reduction of Minimum Rectangle to Strip Packing

Facts:

- Strip packing is NP-hard.
- If there is an efficient α-approximation algorithm for strip packing then for any $\delta > 0$ there is an efficient $(1+\delta)\alpha$-approximation algorithm for finding the minimum area a.p. rectangle for packing a.p. rectangles under translation.

Algorithm:

Let h_0 be the minimum height of the rectangles, H the sum of all heights.

1. Apply the appproximative strip packing algorithm to strips of heights $h_0, \ldots, h_k$, where $h_i = h_0(1+\delta)^i$ and k the smallest integer such that $h_k \geq H$.

Reduction of Minimum Rectangle to Strip Packing

Facts:

- Strip packing is NP-hard.
- If there is an efficient α-approximation algorithm for strip packing then for any $\delta > 0$ there is an efficient $(1+\delta)\alpha$-approximation algorithm for finding the minimum area a.p. rectangle for packing a.p. rectangles under translation.

Algorithm:

Let h_0 be the minimum height of the rectangles, H the sum of all heights.

1. Apply the appproximative strip packing algorithm to strips of heights $h_0, \ldots, h_k$, where $h_i = h_0(1+\delta)^i$ and k the smallest integer such that $h_k \geq H$.
2. Keep track of the rectangle areas obtained in step 1 and return the rectangle and packing achieving the minimum of these areas.

Reduction of Minimum Rectangle to Strip Packing

Facts:

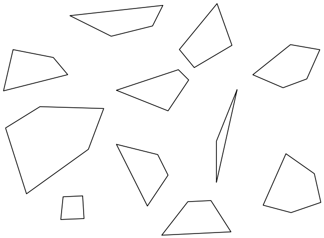
- Strip packing is NP-hard.
- If there is an efficient α-approximation algorithm for strip packing then for any $\delta > 0$ there is an efficient $(1 + \delta)\alpha$-approximation algorithm for finding the minimum area a.p. rectangle for packing a.p. rectangles under translation.
- The best approximation algorithm for strip packing known achieves a factor of $5/3 + \varepsilon$ for any $\varepsilon > 0$ (Harren et al. 2014)
.
- For any $\varepsilon > 0$ there is an $5/3 + \varepsilon$-approximation algorithm for finding the smallest a.p. rectangle for packing a given set of rectangles under translation.

Convex Polygons Under Translation

(Alt, de Berg, Knauer, JOCG 2017)

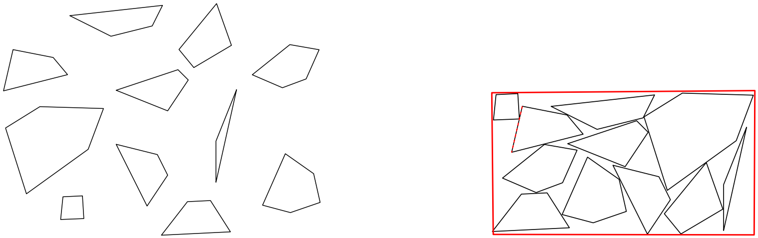
Convex Polygons Under Translation

(Alt, de Berg, Knauer, JOCG 2017)



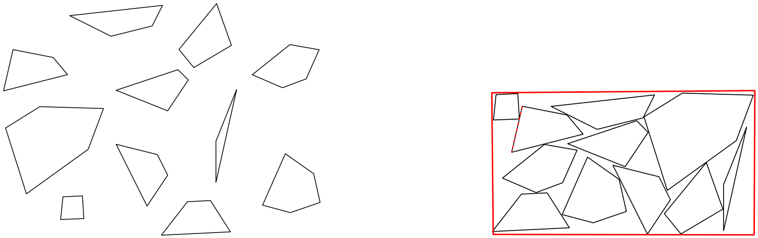
Convex Polygons Under Translation

(Alt, de Berg, Knauer, JOCG 2017)



Convex Polygons Under Translation

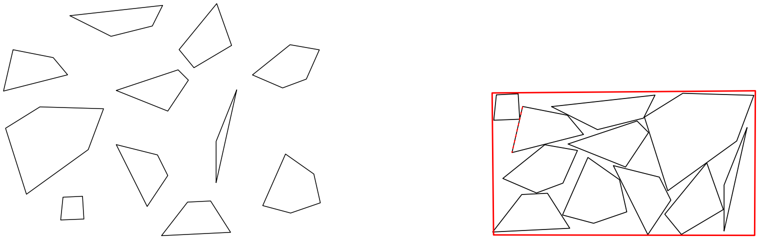
(Alt, de Berg, Knauer, JOCG 2017)



- set P of arbitrary convex polygons

Convex Polygons Under Translation

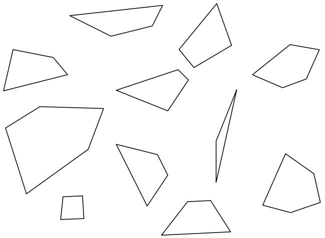
(Alt, de Berg, Knauer, JOCG 2017)

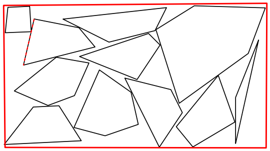


- set P of arbitrary convex polygons
- approximating the minimum area axis-parallel rectangle containing P under translations

Convex Polygons Under Translation

(Alt, de Berg, Knauer, JOCG 2017)





- set P of arbitrary convex polygons
- approximating the minimum area axis-parallel rectangle containing P under translations
- approximating the minimum area convex container for P under translations

Algorithm for axis-parallel rectangular containers

Algorithm for axis-parallel rectangular containers

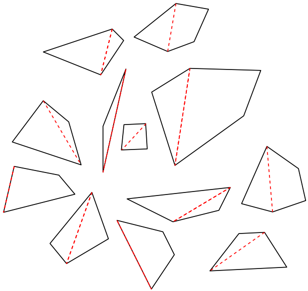
1. group the polygons of P into height classes: $[h_{max}, h_{max}/2), [h_{max}/2, h_{max}/4)...$

Algorithm for axis-parallel rectangular containers

1. group the polygons of P into height classes: $[h_{max}, h_{max}/2), [h_{max}/2, h_{max}/4)...$
2. within each class $i, i = 0, 1, ...$:
   sort the objects by descending slopes ($\in [0, \pi]$) of their spines and place them as far left as possible into an axis-parallel rectangle of height $h_{max}/2^i$

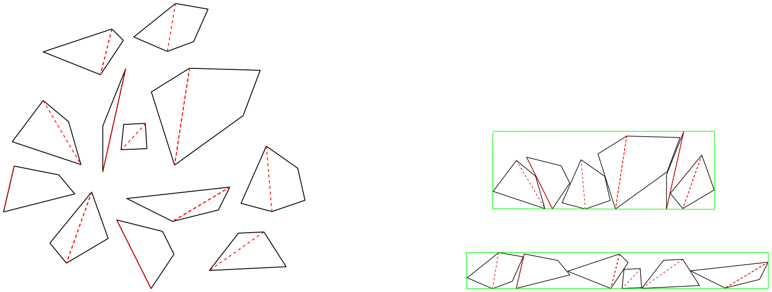
Algorithm for axis-parallel rectangular containers

1. group the polygons of P into height classes:
   $[h_{max}, h_{max}/2), [h_{max}/2, h_{max}/4)\ldots$
2. within each class $i, i = 0, 1, \ldots$:
   sort the objects by descending slopes ($\in [0, \pi]$) of their spines and place them as far left as possible into an axis-parallel rectangle of height $h_{max}/2^i$



Algorithm for axis-parallel rectangular containers

1. group the polygons of P into height classes: $[h_{max}, h_{max}/2), [h_{max}/2, h_{max}/4)\ldots$
2. within each class $i, i = 0, 1, \ldots$: sort the objects by descending slopes ($\in [0, \pi]$) of their spines and place them as far left as possible into an axis-parallel rectangle of height $h_{max}/2^i$



Algorithm for axis-parallel rectangular containers

1. group the polygons of P into height classes:
   $[h_{max}, h_{max}/2), [h_{max}/2, h_{max}/4)\ldots$
2. within each class $i, i = 0, 1, \ldots$:
   sort the objects by descending slopes ($\in [0, \pi]$) of their spines
   and place them as far left as possible into an axis-parallel
   rectangle of height $h_{max}/2^i$
3. cut the small boxes for the different height-classes into
   rectangles of equal width $2w_{max}$, stack all these rectangles,
   and return the resulting rectangle

Constant factor approximation: proof idea

important observations about the optimal packing area:

Constant factor approximation: proof idea

important observations about the optimal packing area:

1. $OPT \geq$ the total area of the polygons

Constant factor approximation: proof idea

important observations about the optimal packing area:

1. $OPT \geq$ the total area of the polygons
2. $OPT \geq h_{max} \cdot w_{max}$

Constant factor approximation: proof idea

important observations about the optimal packing area:

1. $OPT \geq$ the total area of the polygons
2. $OPT \geq h_{max} \cdot w_{max}$

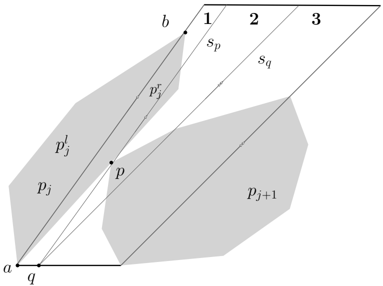
area used in step 2:

Constant factor approximation: proof idea

important observations about the optimal packing area:

1. $OPT \geq$ the total area of the polygons
2. $OPT \geq h_{max} \cdot w_{max}$

area used in step 2:

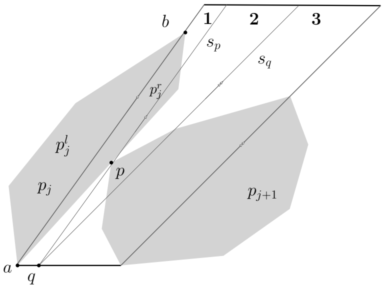


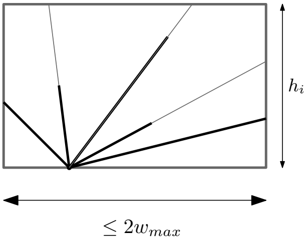
Constant factor approximation: proof idea

important observations about the optimal packing area:

1. $OPT \geq$ the total area of the polygons
2. $OPT \geq h_{max} \cdot w_{max}$

area used in step 2:





Algorithm for axis-parallel rectangular containers

1. group the polygons of P into height classes:
   $[h_{max}, h_{max}/2), [h_{max}/2, h_{max}/4)\ldots$
2. within each class $i, i = 0, 1, \ldots$:
   sort the objects by descending slopes ($\in [0, \pi]$) of their spines and place them as far left as possible into an axis-parallel rectangle of height $h_{max}/2^i$
3. cut the small boxes for the different height-classes into rectangles of equal width $2w_{max}$ and stack all these rectangles, and return the resulting rectangle

Algorithm for axis-parallel rectangular containers

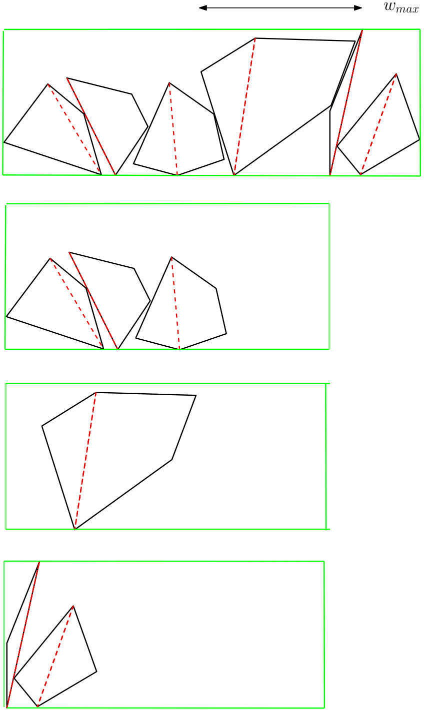
1. group the polygons of P into height classes:
   $[h_{max}, h_{max} \cdot \alpha), [h_{max} \cdot \alpha, h_{max} \cdot \alpha^2)$...
2. within each class $i, i = 0, 1, ...$:
   sort the objects by descending slopes ($\in [0, \pi]$) of their spines and place them as far left as possible into an axis-parallel rectangle of height $h_{max} \cdot \alpha^i$
3. cut the small boxes for the different height-classes into rectangles of equal width cw_{max}, and stack all these rectangles, and return the resulting rectangle

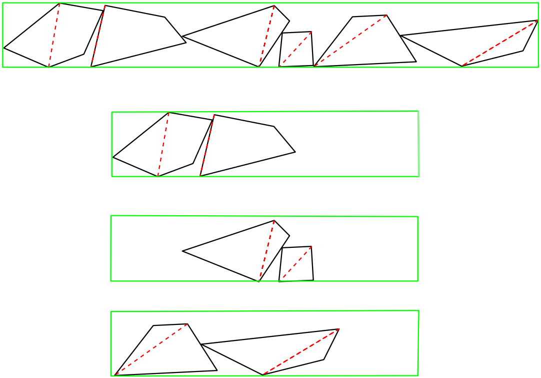
Algorithm for axis-parallel rectangular containers

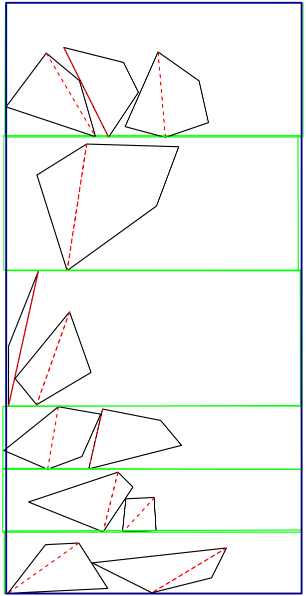
1. group the polygons of P into height classes: $[h_{max}, h_{max} \cdot \alpha), [h_{max} \cdot \alpha, h_{max} \cdot \alpha^2)$...
2. within each class $i, i = 0, 1, \ldots$:
   sort the objects by descending slopes ($\in [0, \pi]$) of their spines and place them as far left as possible into an axis-parallel rectangle of height $h_{max} \cdot \alpha^i$
3. cut the small boxes for the different height-classes into rectangles of equal width $c w_{max}$, and stack all these rectangles, and return the resulting rectangle

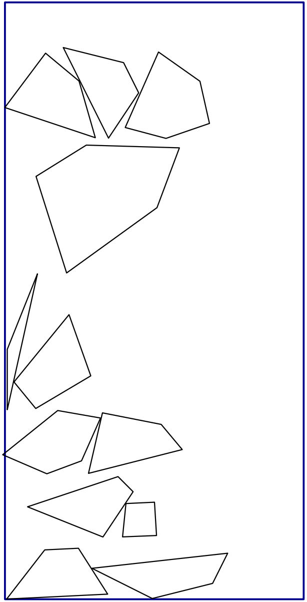
gives an approximation factor of $f(\alpha, c) = \left(1 + \frac{1}{c}\right) \cdot \frac{2+c\alpha}{\alpha-\alpha^2}$

this is optimal for $\alpha = 0.407..$, $c = 2.214..$, namely 17.449...


w_{max}







Algorithm for arbitrary convex containers

Algorithm for arbitrary convex containers

1. Find an orientation $\phi \in S^1$ which minimizes $h_{max}(\phi)w_{max}(\phi)$.

Algorithm for arbitrary convex containers

1. Find an orientation $\phi \in S^1$ which minimizes $h_{max}(\phi) w_{max}(\phi)$.
2. Use the previous algorithm with respect to orientation ϕ.

Results

For packing a set of convex polygons using translations:

Results

For packing a set of convex polygons using translations:

1. An approximation to the minimum area axis parallel rectangle can be found in $O(n \log n)$ time.

Results

For packing a set of convex polygons using translations:

1. An approximation to the minimum area axis parallel rectangle can be found in $O(n \log n)$ time.
2. An approximation to the minimum area convex container can be found in $O(n \log n)$ time.

Results

For packing a set of convex polygons using translations:

1. An approximation to the minimum area axis parallel rectangle can be found in $O(n \log n)$ time. Approximation factor: 18
2. An approximation to the minimum area convex container can be found in $O(n \log n)$ time.

Results

For packing a set of convex polygons using translations:

1. An approximation to the minimum area axis parallel rectangle can be found in $O(n \log n)$ time. Approximation factor: 18
2. An approximation to the minimum area convex container can be found in $O(n \log n)$ time. Approximation factor: $16 + 8\sqrt{3} = 29.86..$