Polynomiality for Bin Packing with a Constant Number of Item Types

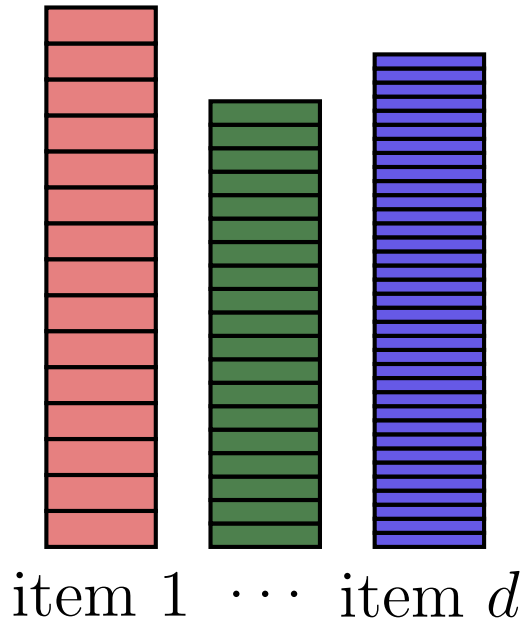
Michel X. Goemans & Thomas Rothvoß

MIP 2013, Madison

Massachusetts
Institute of
Technology

Bin Packing / Cutting Stock

Input:

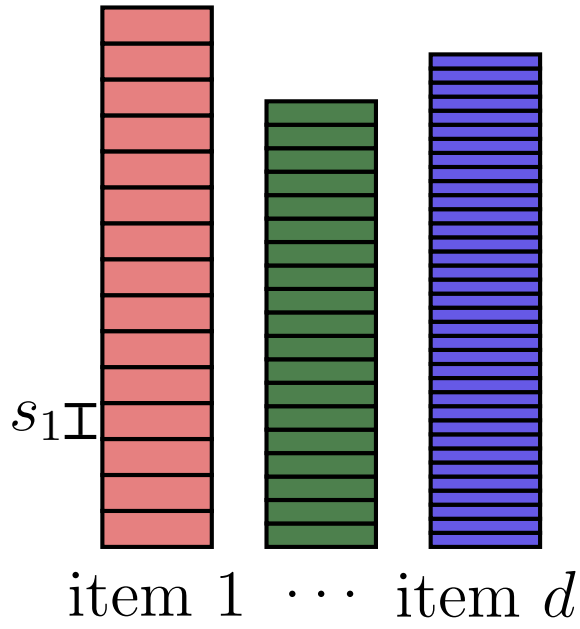


Bin Packing / Cutting Stock

Input:

- Item sizes $s_1, \ldots, s_d \in [0, 1]$

Input:

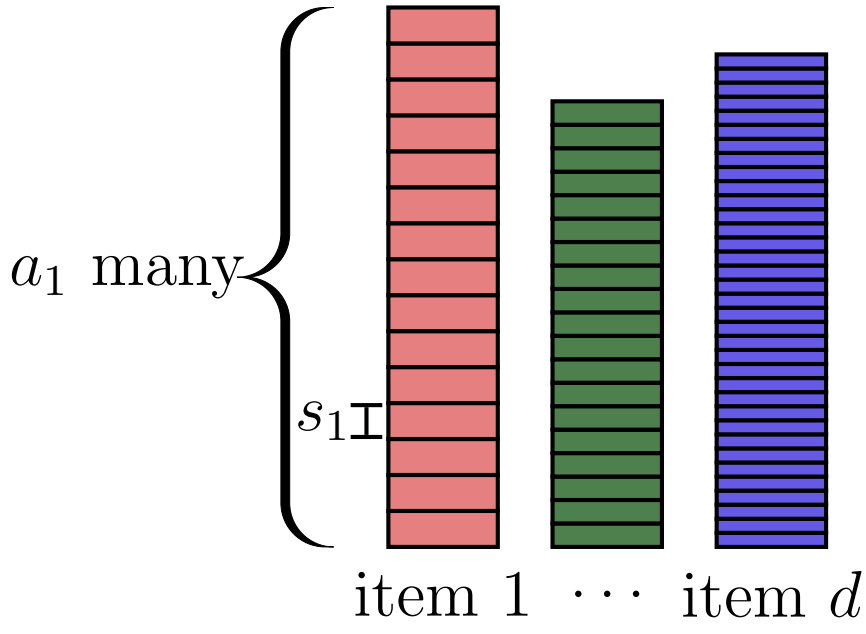


Bin Packing / Cutting Stock

Input:

- Item sizes $s_1, \ldots, s_d \in [0, 1]$
- Multiplicities $a_1, \ldots, a_d \in \mathbb{N}$

Input:



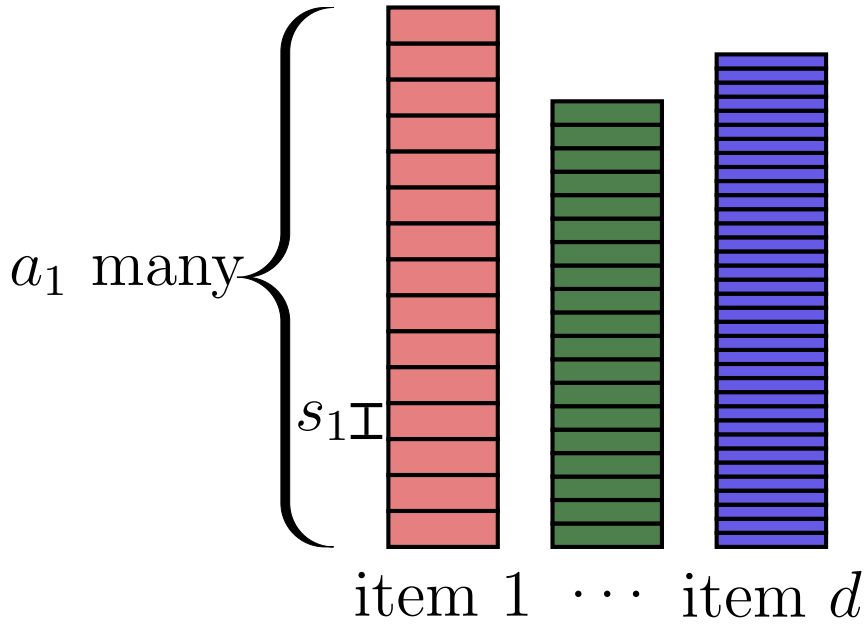
Bin Packing / Cutting Stock

Input:

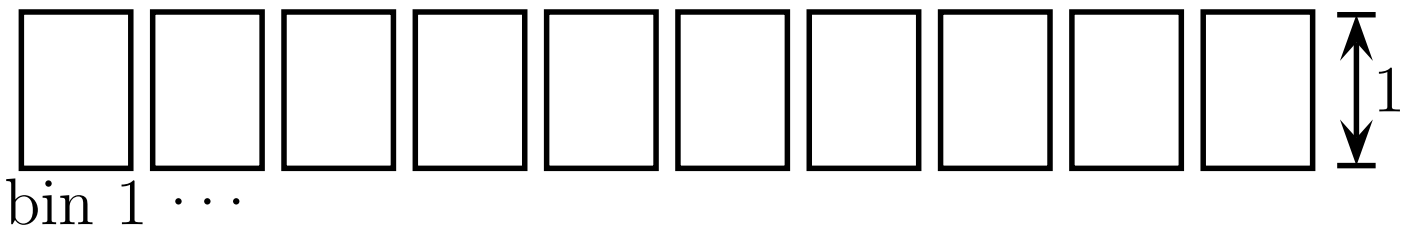
- Item sizes $s_1, \ldots, s_d \in [0, 1]$
- Multiplicities $a_1, \ldots, a_d \in \mathbb{N}$

Goal: Pack items into minimum number of **bins** of size 1.

Input:



Solution:



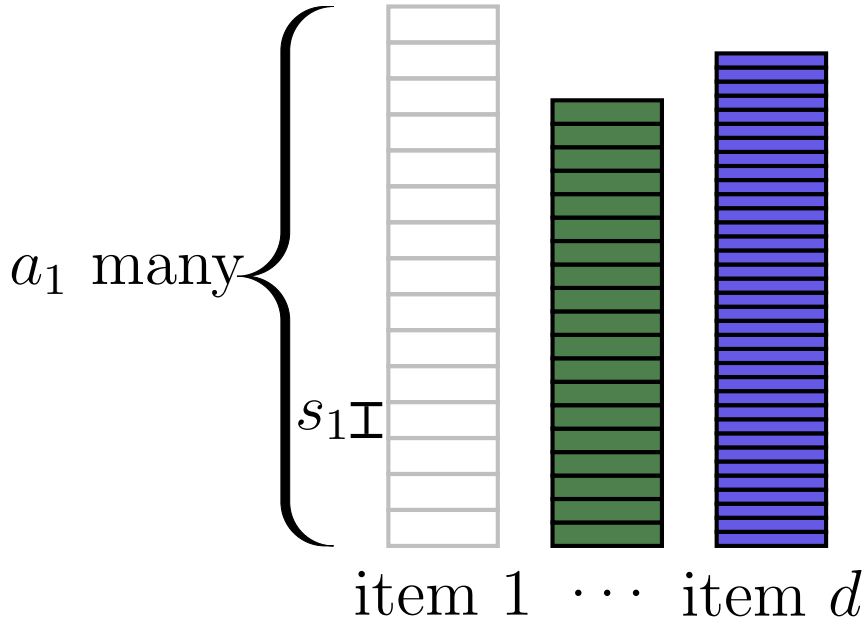
Bin Packing / Cutting Stock

Input:

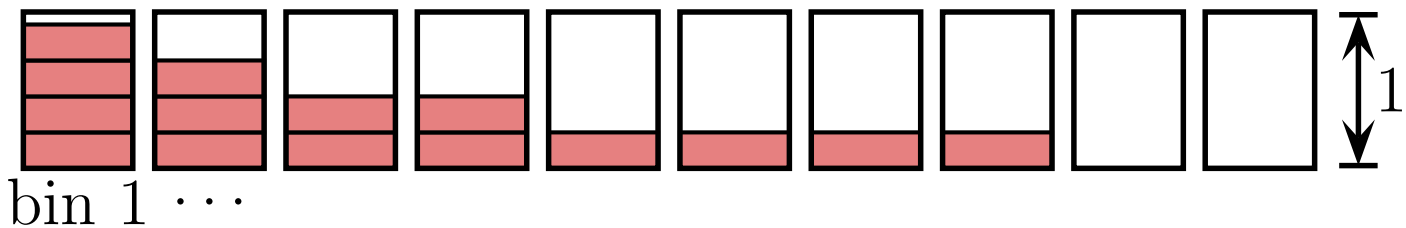
- Item sizes $s_1, \ldots, s_d \in [0, 1]$
- Multiplicities $a_1, \ldots, a_d \in \mathbb{N}$

Goal: Pack items into minimum number of **bins** of size 1.

Input:



Solution:



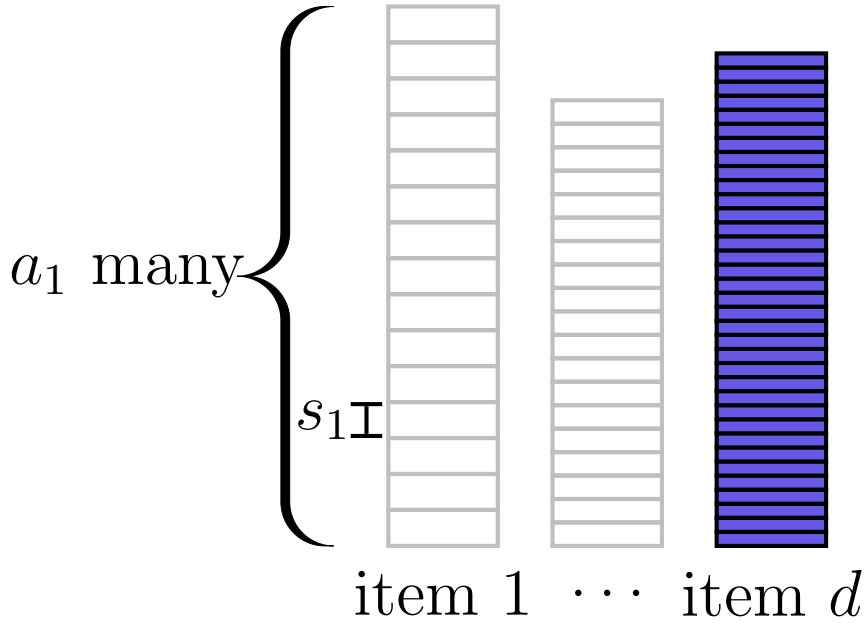
Bin Packing / Cutting Stock

Input:

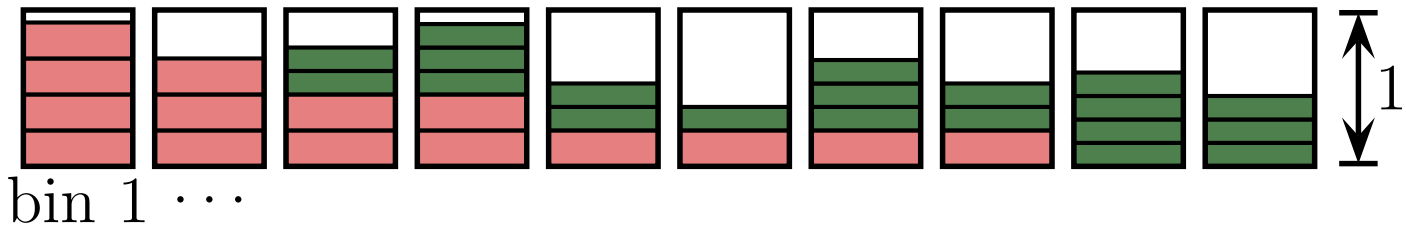
- Item sizes $s_1, \ldots, s_d \in [0, 1]$
- Multiplicities $a_1, \ldots, a_d \in \mathbb{N}$

Goal: Pack items into minimum number of **bins** of size 1.

Input:



Solution:



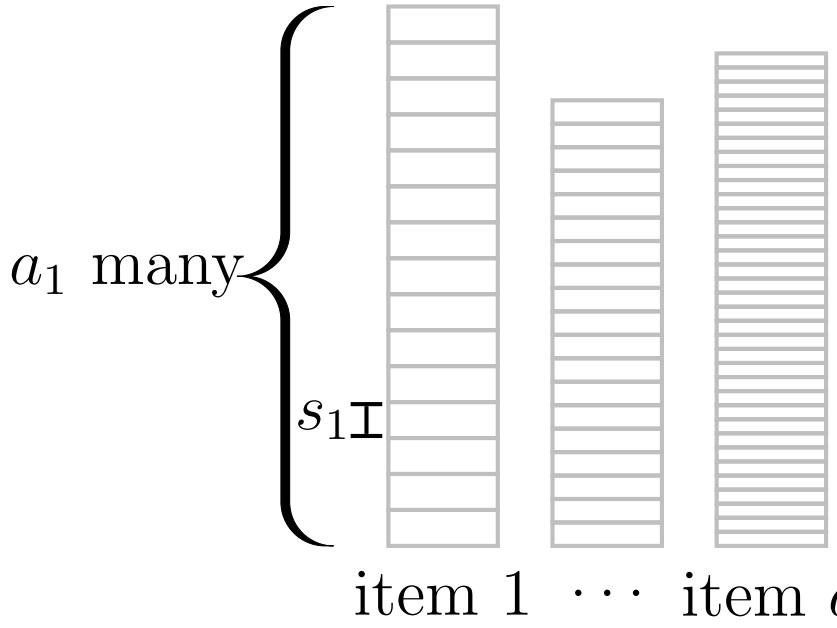
Bin Packing / Cutting Stock

Input:

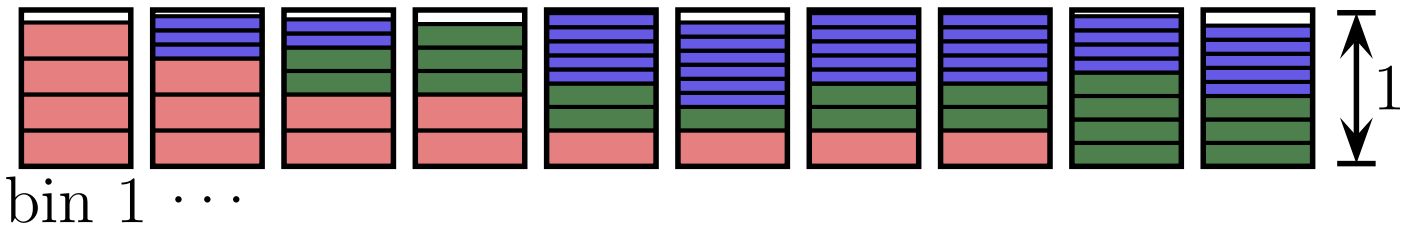
- Item sizes $s_1, \ldots, s_d \in [0, 1]$
- Multiplicities $a_1, \ldots, a_d \in \mathbb{N}$

Goal: Pack items into minimum number of **bins** of size 1.

Input:



Solution:



Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$ [Garey & Johnson '79]

Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$ [Garey & Johnson '79]
- Asymptotic FPTAS
  $OPT + O(\log^2 d)$ [Karmarkar & Karp '82]
  $OPT + O(\log d \cdot \log\log d)$ [R. '13]
  (running time poly$(\sum_{i=1}^{d} a_i)$)

Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$ [Garey & Johnson '79]
- Asymptotic FPTAS
  $OPT + O(\log^2 d)$ [Karmarkar & Karp '82]
  $OPT + O(\log d \cdot \log\log d)$ [R. '13]
  (running time poly$(\sum_{i=1}^{d} a_i)$)
- $\in$ **NP** [Eisenbrand & Shmonin '06]

Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$ [Garey & Johnson '79]
- Asymptotic FPTAS
  $OPT + O(\log^2 d)$ [Karmarkar & Karp '82]
  $OPT + O(\log d \cdot \log\log d)$ [R. '13]
  (running time poly$(\sum_{i=1}^{d} a_i)$)
- $\in$ **NP** [Eisenbrand & Shmonin '06]

For constant d:

Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$
  [Garey & Johnson '79]
- Asymptotic FPTAS
  $OPT + O(\log^2 d)$ [Karmarkar & Karp '82]
  $OPT + O(\log d \cdot \log\log d)$ [R. '13]
  (running time poly$(\sum_{i=1}^{d} a_i)$)
- $\in$ **NP** [Eisenbrand & Shmonin '06]

For constant d:

- Polytime for $d = 2$ [McCormick, Smallwood, Spieksma '97]

Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$ [Garey & Johnson '79]
- Asymptotic FPTAS
  $OPT + O(\log^2 d)$ [Karmarkar & Karp '82]
  $OPT + O(\log d \cdot \log\log d)$ [R. '13]
  (running time poly$(\sum_{i=1}^{d} a_i)$)
- $\in$ **NP** [Eisenbrand & Shmonin '06]

For constant d:

- Polytime for $d = 2$ [McCormick, Smallwood, Spieksma '97]
- $OPT + \mathbf{1}$ in time $2^{2^{O(d)}} \cdot$ poly [Jansen & Solis-Oba '10]

Polynomial time algorithms

For general d:

- **NP**-hard to distinguish $OPT \leq 2$ or $OPT \geq 3$ [Garey & Johnson '79]
- Asymptotic FPTAS
  $OPT + O(\log^2 d)$ [Karmarkar & Karp '82]
  $OPT + O(\log d \cdot \log\log d)$ [R. '13]
  (running time poly$(\sum_{i=1}^{d} a_i)$)
- $\in$ **NP** [Eisenbrand & Shmonin '06]

For constant d:

- Polytime for $d = 2$ [McCormick, Smallwood, Spieksma '97]
- $OPT + \mathbf{1}$ in time $2^{2^{O(d)}} \cdot$ poly [Jansen & Solis-Oba '10]

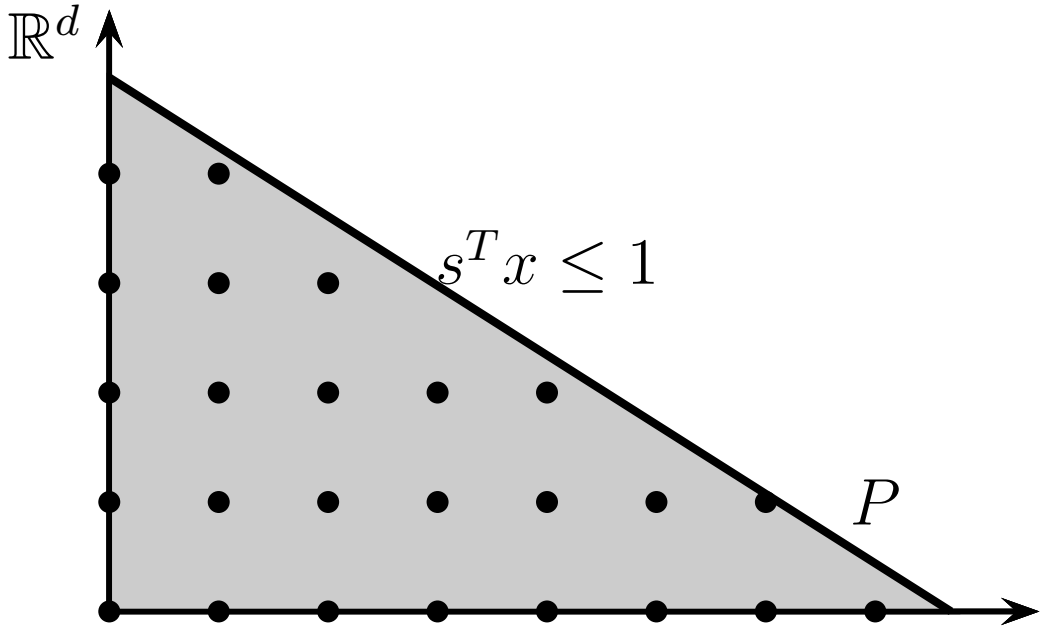
Open problem [ES'06, MSS'97, F'07]

Solvable in poly-time for $d = 3$?

A geometric view

- Define $P = \{x \in \mathbb{R}^d_{\geq 0} \mid s^T x \leq 1\}$

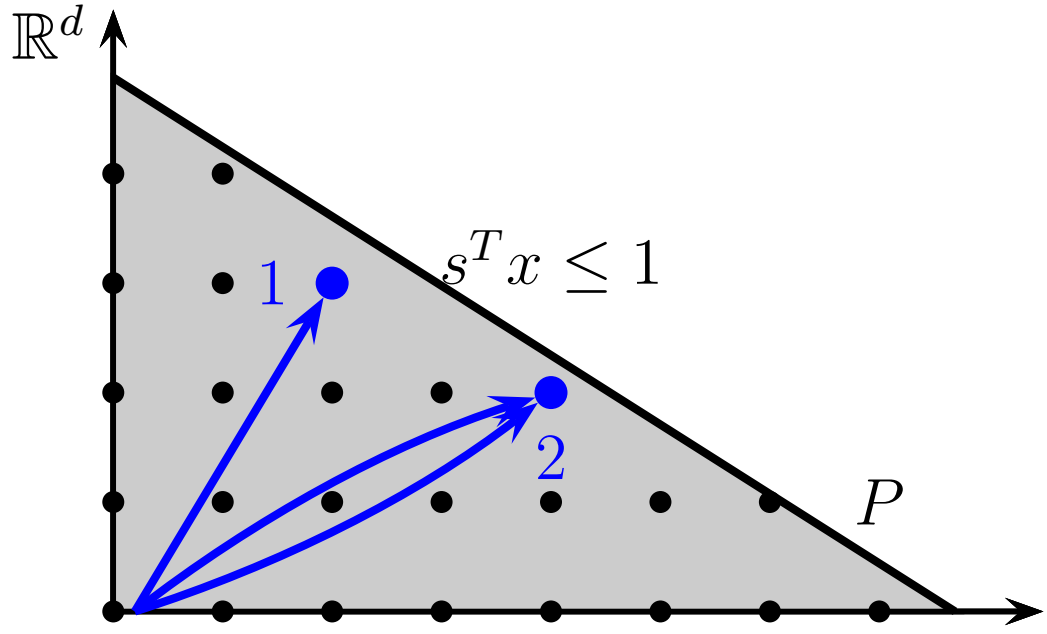
● a



A geometric view

- Define $P = \{x \in \mathbb{R}^d_{\geq 0} \mid s^T x \leq 1\}$

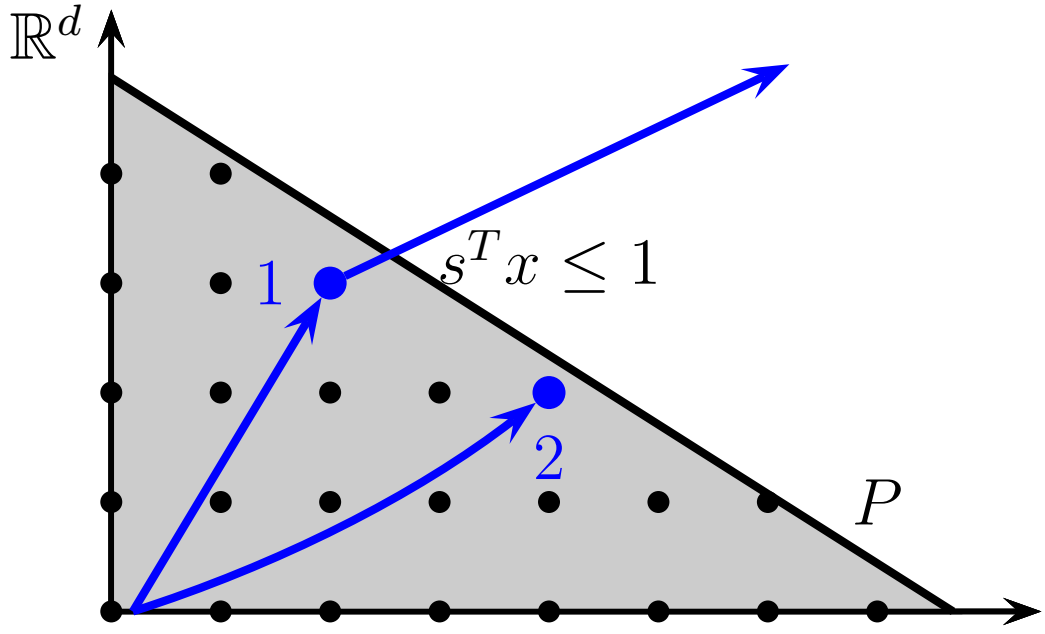
● a



A geometric view

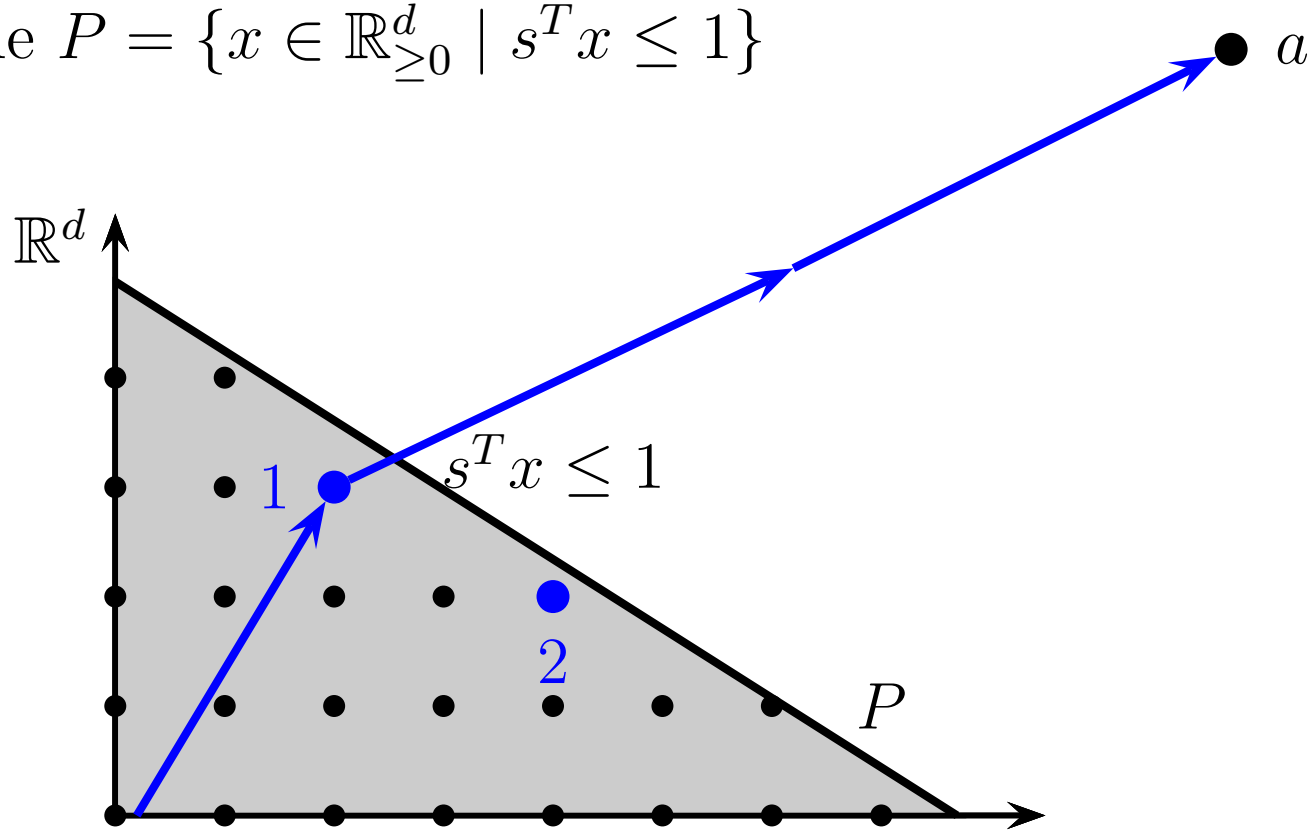
- Define $P = \{x \in \mathbb{R}^d_{\geq 0} \mid s^T x \leq 1\}$

● a



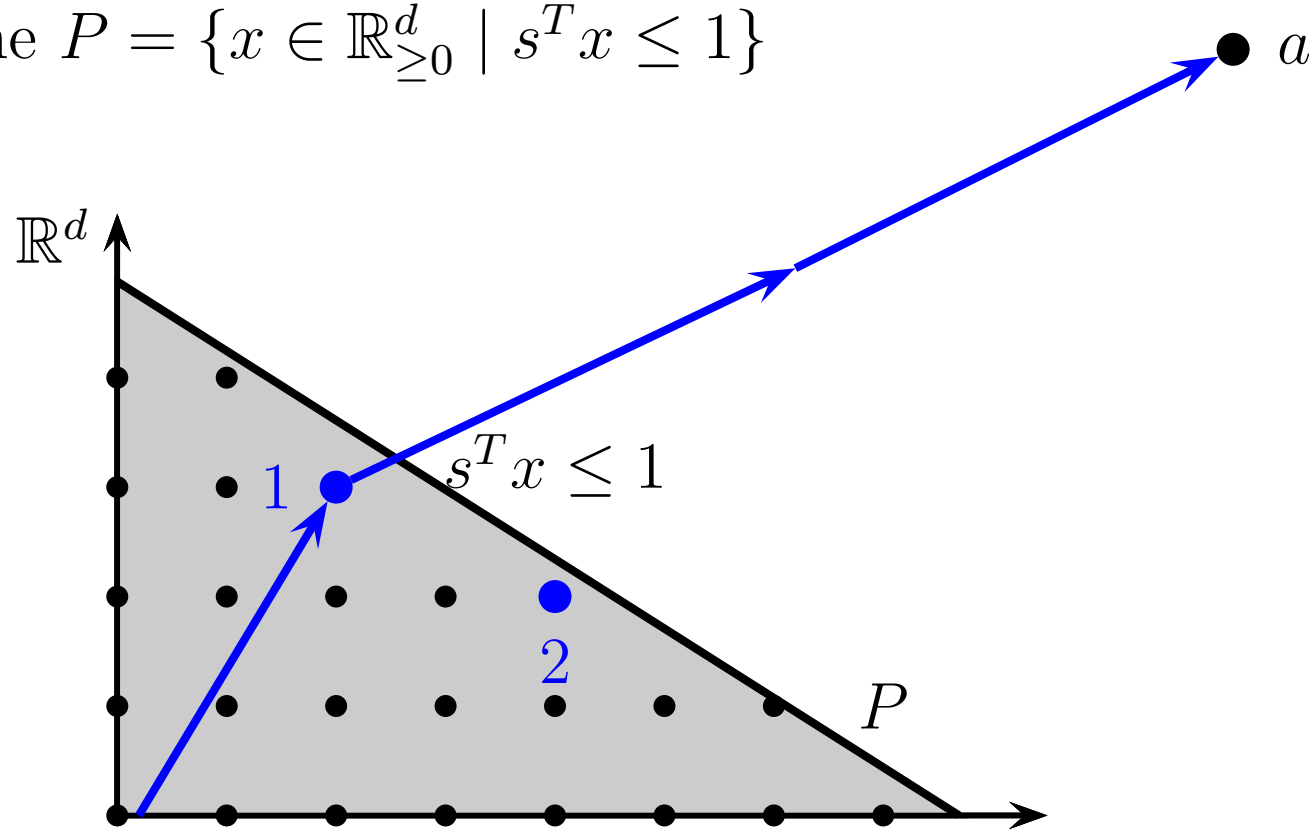
A geometric view

- Define $P = \{x \in \mathbb{R}^d_{\geq 0} \mid s^T x \leq 1\}$



A geometric view

- Define $P = \{x \in \mathbb{R}^d_{\geq 0} \mid s^T x \leq 1\}$

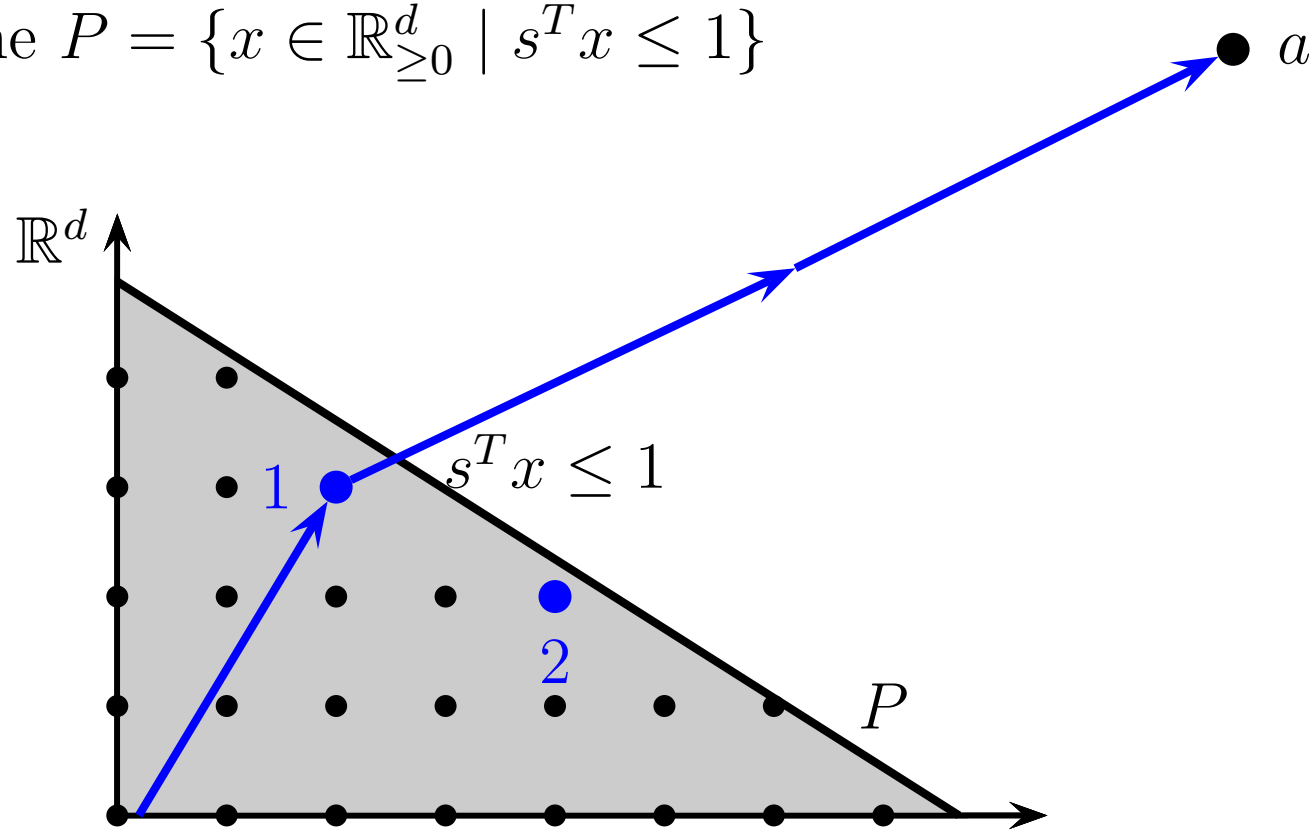


Problems:

- Points in P **exponentially** many

A geometric view

- Define $P = \{x \in \mathbb{R}^d_{\geq 0} \mid s^T x \leq 1\}$



Problems:

- Points in P **exponentially** many
- Weights can be **exponential**

Main results

Theorem (Goemans, R. '13)

Bin Packing with $d = O(1)$ item sizes can be solved in **poly-time**.

Solves question by

- [McCormick, Smallwood, Spieksma '97]: "*might be* **NP**-*hard for* $d = 3$"
- [Eisenbrand & Shmonin '06]
- [Filippi '07]: "*hard open problem for general* d"

Main results (2)

- **Def.:** $\text{int.cone}(X) := \{\sum_{x \in X} \lambda_x \cdot x \mid \lambda_x \in \mathbb{Z}_{\geq 0}\}$

Main results (2)

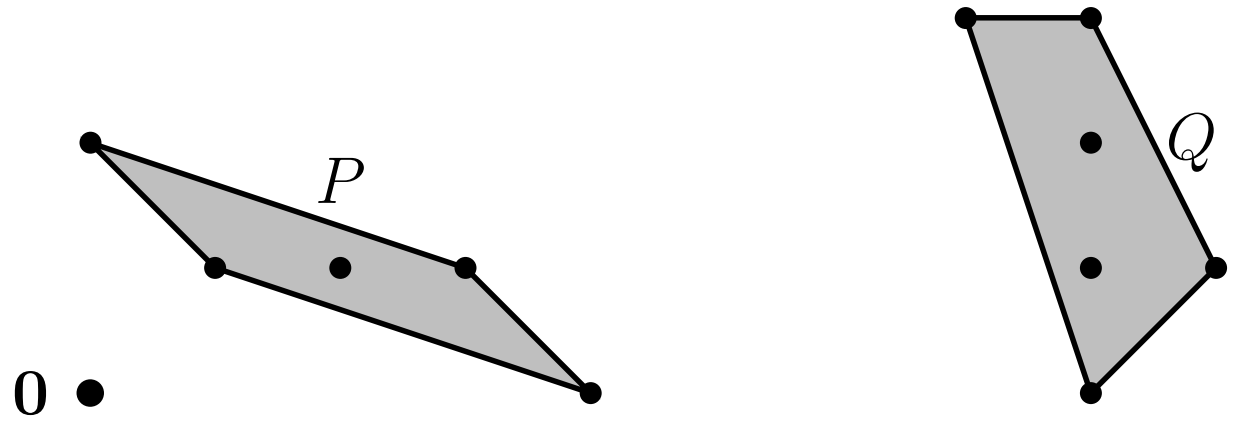
- **Def.:** $\text{int.cone}(X) := \{\sum_{x \in X} \lambda_x \cdot x \mid \lambda_x \in \mathbb{Z}_{\geq 0}\}$

Theorem (Goemans, R. '13)

For **fixed-dim. polytopes** $P, Q \subseteq \mathbb{R}^d$, testing

$$\text{int.cone}(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$$

is doable in **poly-time** (actually $\text{inputlength}^{2^{O(d)}}$).



Main results (2)

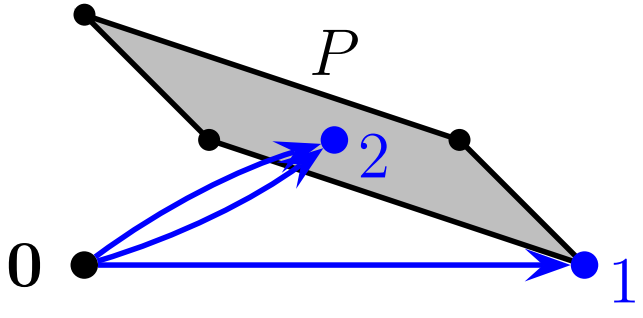
- **Def.:** $\text{int.cone}(X) := \{\sum_{x \in X} \lambda_x \cdot x \mid \lambda_x \in \mathbb{Z}_{\geq 0}\}$

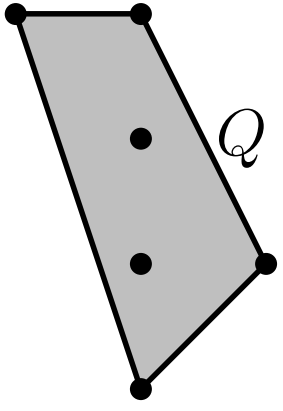
Theorem (Goemans, R. '13)

For **fixed-dim. polytopes** $P, Q \subseteq \mathbb{R}^d$, testing

$$\text{int.cone}(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$$

is doable in **poly-time** (actually $\text{inputlength}^{2^{O(d)}}$).





Main results (2)

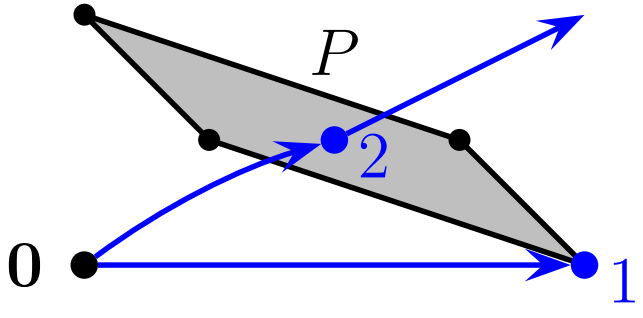
- **Def.:** $\text{int.cone}(X) := \{\sum_{x \in X} \lambda_x \cdot x \mid \lambda_x \in \mathbb{Z}_{\geq 0}\}$

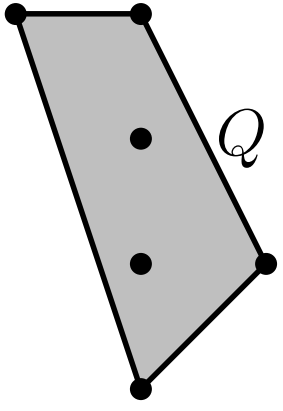
Theorem (Goemans, R. '13)

For **fixed-dim. polytopes** $P, Q \subseteq \mathbb{R}^d$, testing

$$\text{int.cone}(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$$

is doable in **poly-time** (actually $\text{inputlength}^{2^{O(d)}}$).





Main results (2)

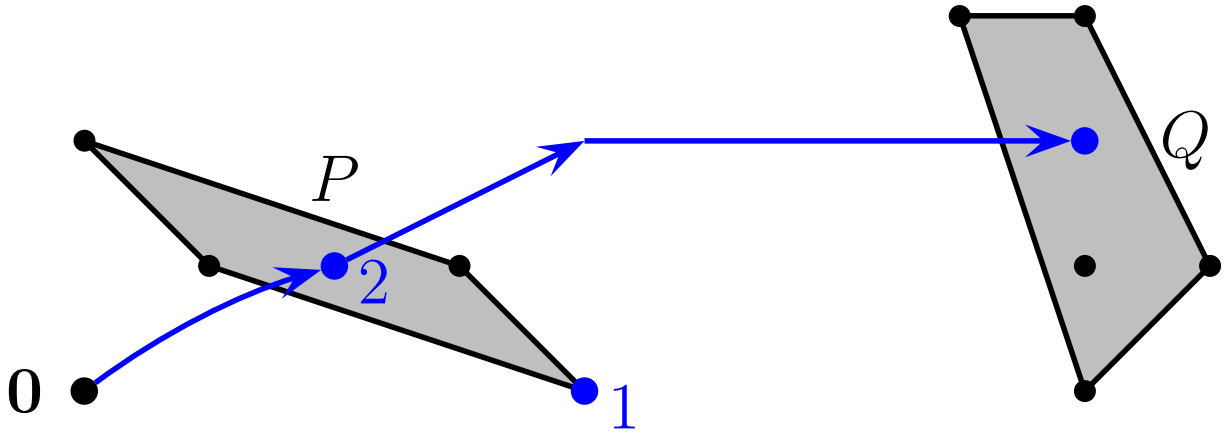
- **Def.:** $\text{int.cone}(X) := \{\sum_{x \in X} \lambda_x \cdot x \mid \lambda_x \in \mathbb{Z}_{\geq 0}\}$

Theorem (Goemans, R. '13)

For **fixed-dim. polytopes** $P, Q \subseteq \mathbb{R}^d$, testing

$$\text{int.cone}(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$$

is doable in **poly-time** (actually $\text{inputlength}^{2^{O(d)}}$).



Main results (2)

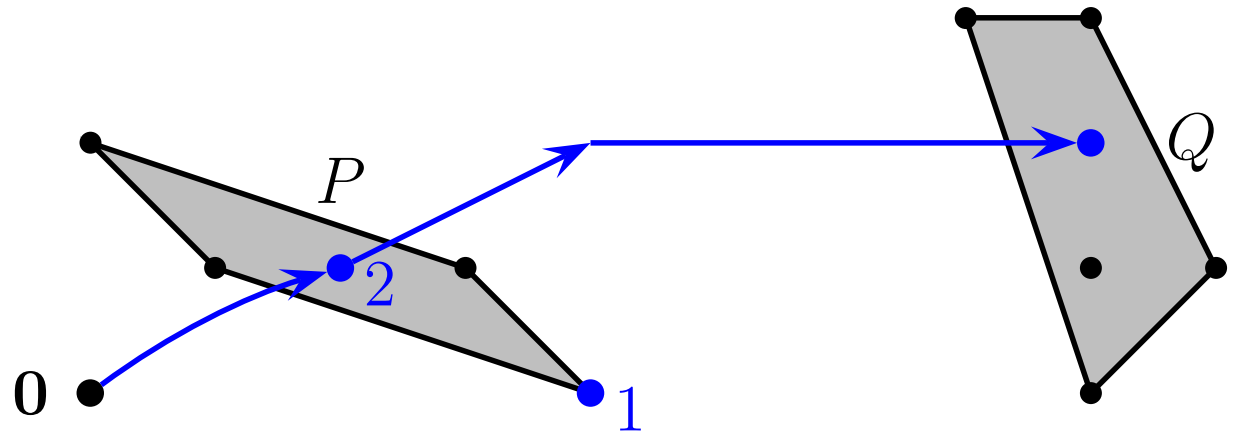
- **Def.:** $\text{int.cone}(X) := \{\sum_{x \in X} \lambda_x \cdot x \mid \lambda_x \in \mathbb{Z}_{\geq 0}\}$

Theorem (Goemans, R. '13)

For **fixed-dim. polytopes** $P, Q \subseteq \mathbb{R}^d$, testing

$$\text{int.cone}(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$$

is doable in **poly-time** (actually $\text{inputlength}^{2^{O(d)}}$).



- For Bin Packing:
  $P := \{\binom{x}{1} \mid s^T x \leq 1,\ x \geq \mathbf{0}\}$ and $Q := \{\binom{a}{OPT}\}$

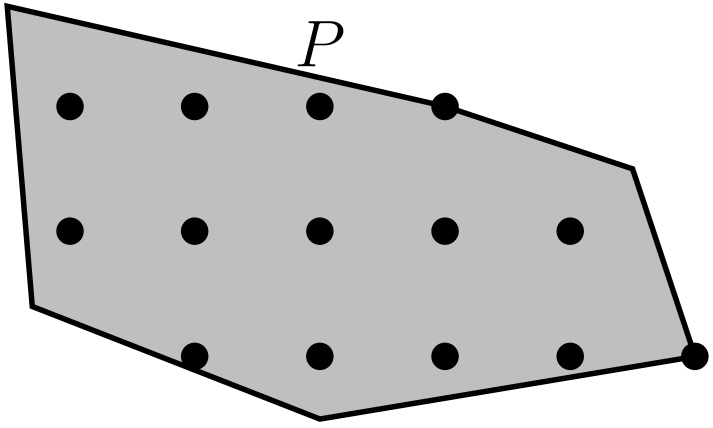
Int. conic combinations

Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

needs at most 2^d points.



Int. conic combinations

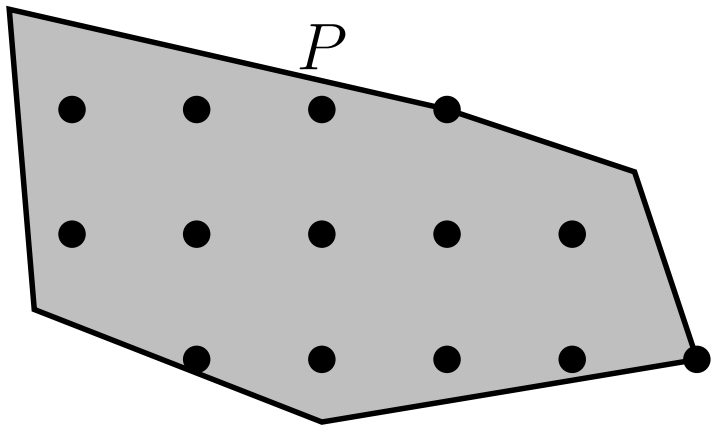
Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

needs at most 2^d points.

- Suppose $|\text{supp}(\lambda)| > 2^d$



Int. conic combinations

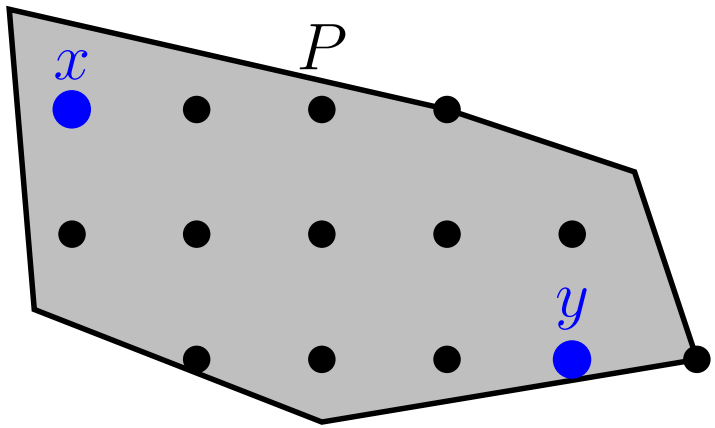
Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

needs at most 2^d points.

- Suppose $|\text{supp}(\lambda)| > 2^d$
- Take points $x, y \in \text{supp}(\lambda)$ of **same parity**



Int. conic combinations

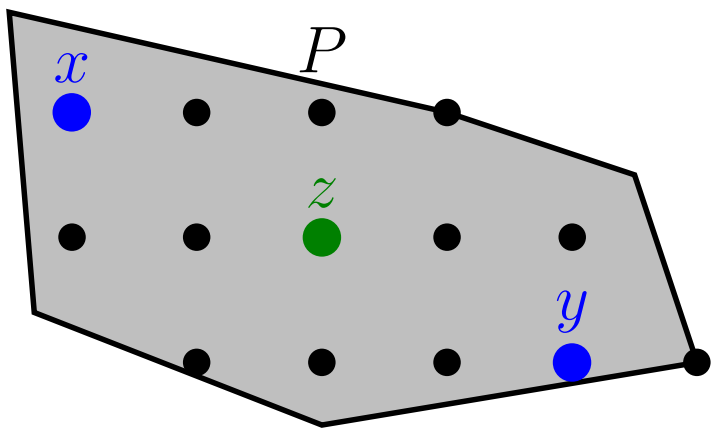
Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

needs at most 2^d points.

- Suppose $|\text{supp}(\lambda)| > 2^d$
- Take points $x, y \in \text{supp}(\lambda)$ of **same parity**
- Midpoint $z = \frac{1}{2}(x + y) \in P \cap \mathbb{Z}^d$



Int. conic combinations

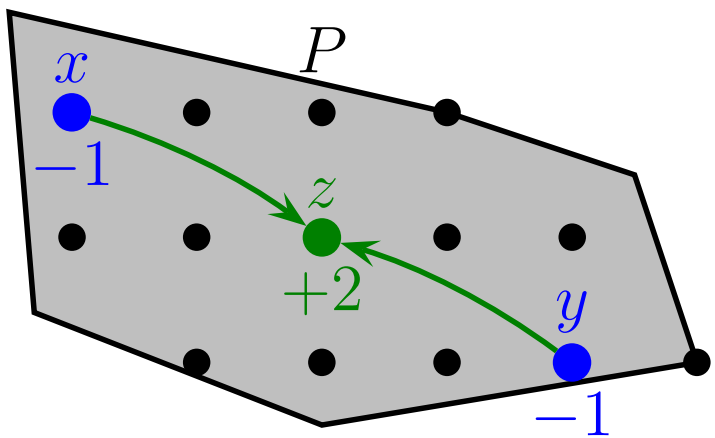
Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

needs at most 2^d points.

- Suppose $|\mathrm{supp}(\lambda)| > 2^d$
- Take points $x, y \in \mathrm{supp}(\lambda)$ of **same parity**
- Midpoint $z = \frac{1}{2}(x + y) \in P \cap \mathbb{Z}^d$
- Move weight from x, y to z



Int. conic combinations

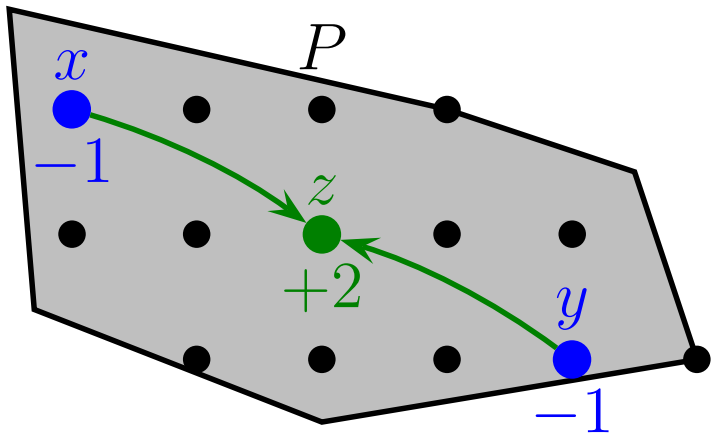
Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

needs at most 2^d points.

- Suppose $|\mathrm{supp}(\lambda)| > 2^d$
- Take points $x, y \in \mathrm{supp}(\lambda)$ of **same parity**
- Midpoint $z = \frac{1}{2}(x + y) \in P \cap \mathbb{Z}^d$
- Move weight from x, y to z
- **Potential function** $\sum_x \lambda_x f(x)$ decreases (f strictly convex) □



Int. conic combinations

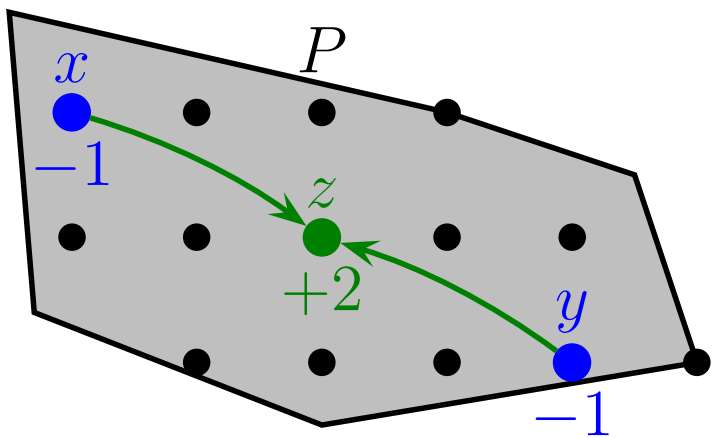
Theorem (Eisenbrand & Shmonin '06)

If $P \subseteq \mathbb{R}^d$ convex, then any integer conic combination

$$a = \sum_{x \in P \cap \mathbb{Z}^d} \lambda_x \cdot x$$

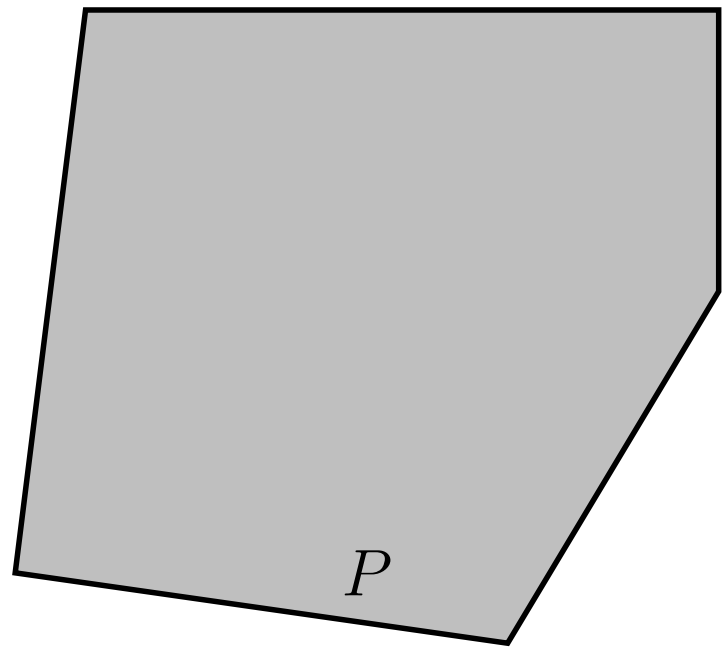
needs at most 2^d points.

- Suppose $|\mathrm{supp}(\lambda)| > 2^d$
- Take points $x, y \in \mathrm{supp}(\lambda)$ of **same parity**
- Midpoint $z = \frac{1}{2}(x + y) \in P \cap \mathbb{Z}^d$
- Move weight from x, y to z
- **Potential function** $\sum_x \lambda_x f(x)$ decreases (f strictly convex) □

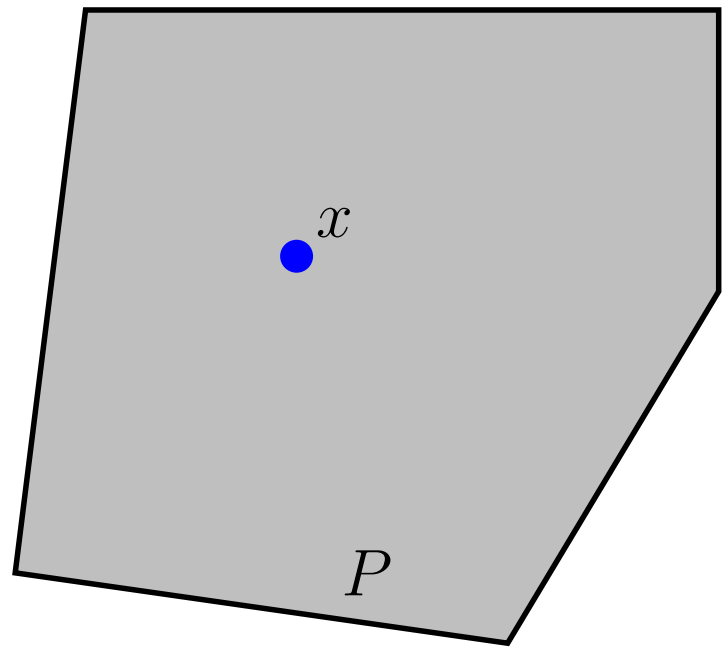


- **Problem:** Still don't know which points to take!

Redistributing weight

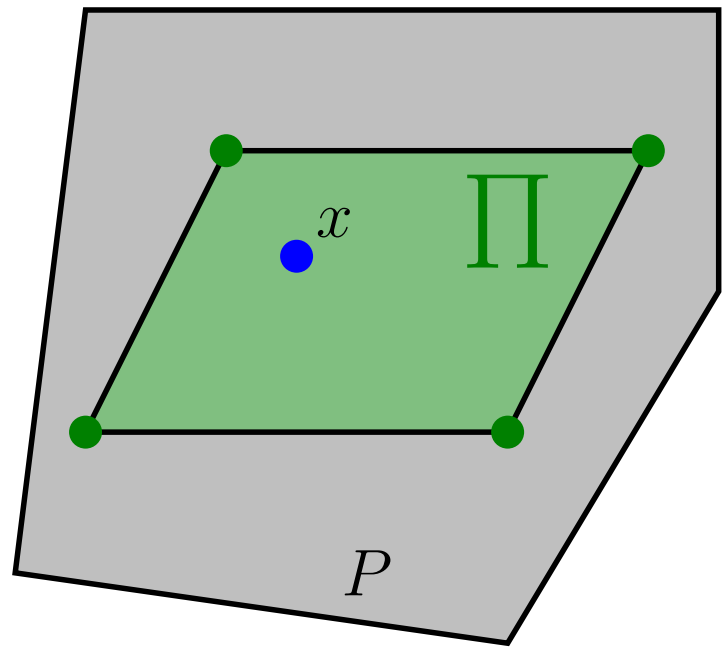


Redistributing weight



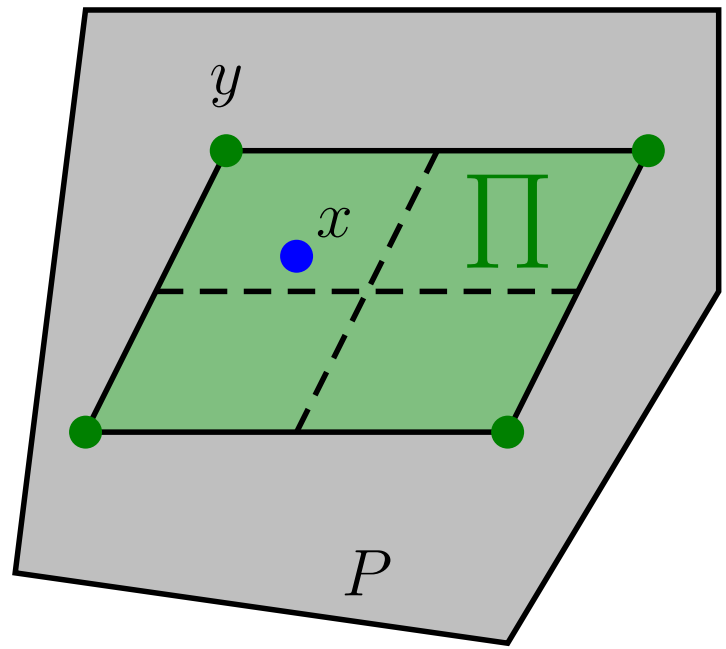
Redistributing weight

- Consider **parallelepiped** $\Pi \ni x$ with integral vertices



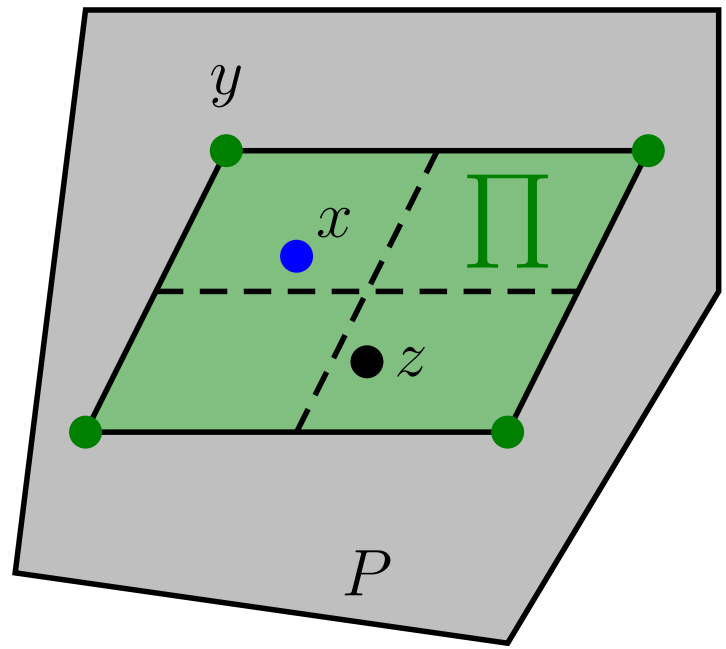
Redistributing weight

- Consider **parallelepiped** $\Pi \ni x$ with integral vertices
- Let y vertex of Π, in quadrant of x



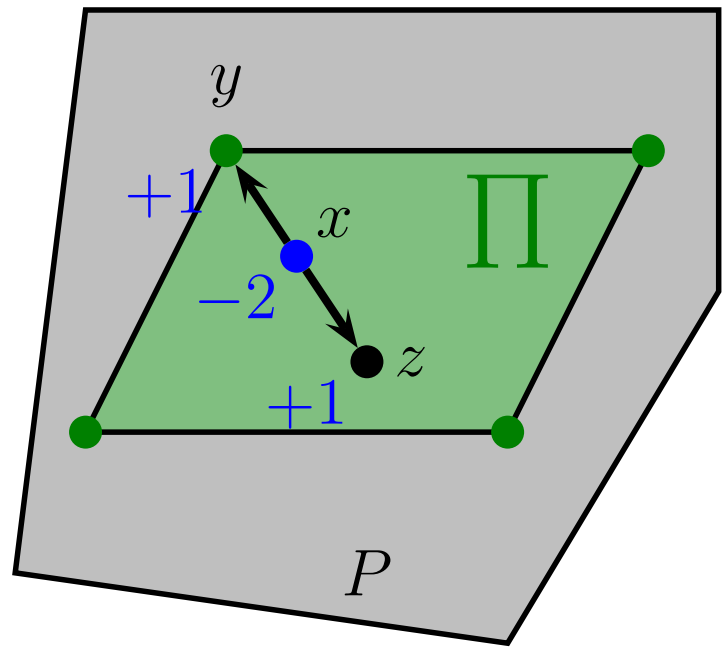
Redistributing weight

- Consider **parallelepiped** $\Pi \ni x$ with integral vertices
- Let y vertex of Π, in quadrant of x
- Let $z \in \Pi \cap \mathbb{Z}^d$ be mirrored point



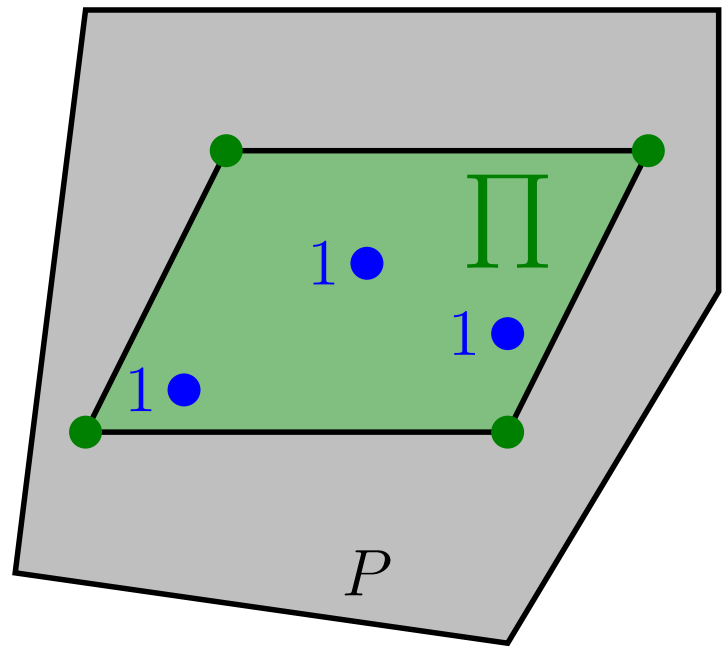
Redistributing weight

- Consider **parallelepiped** $\Pi \ni x$ with integral vertices
- Let y vertex of Π, in quadrant of x
- Let $z \in \Pi \cap \mathbb{Z}^d$ be mirrored point
- If $\lambda_x \geq 2 \Rightarrow$ redistribute weight



Redistributing weight

- Consider **parallelepiped** $\Pi \ni x$ with integral vertices
- Let y vertex of Π, in quadrant of x
- Let $z \in \Pi \cap \mathbb{Z}^d$ be mirrored point
- If $\lambda_x \geq 2 \Rightarrow$ redistribute weight
- At most 2^d points left inside Π



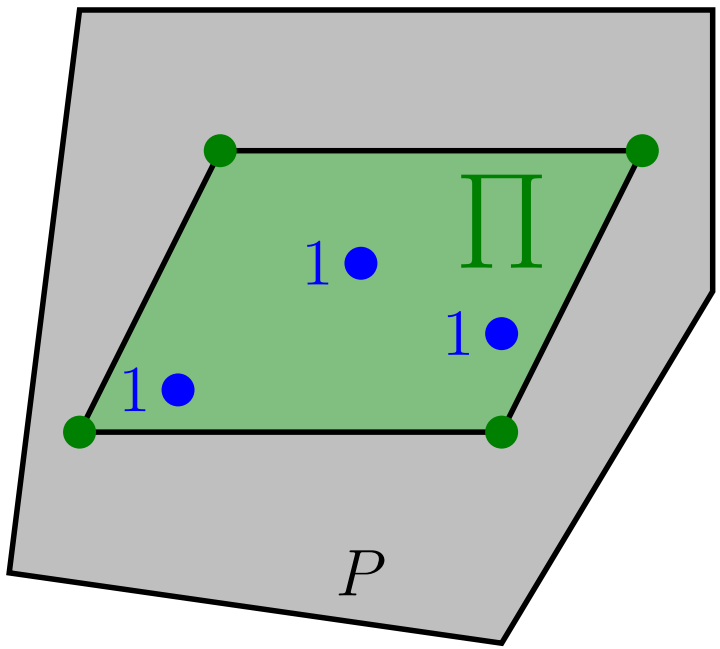
Redistributing weight

Lemma

For x in parallelepiped Π and $\lambda_x \in \mathbb{N}$, one can write

$$\lambda_x x = \text{int.cone(vertices of } \Pi) + \sum \text{ of } 2^d \text{ points in } \Pi \cap \mathbb{Z}^d$$

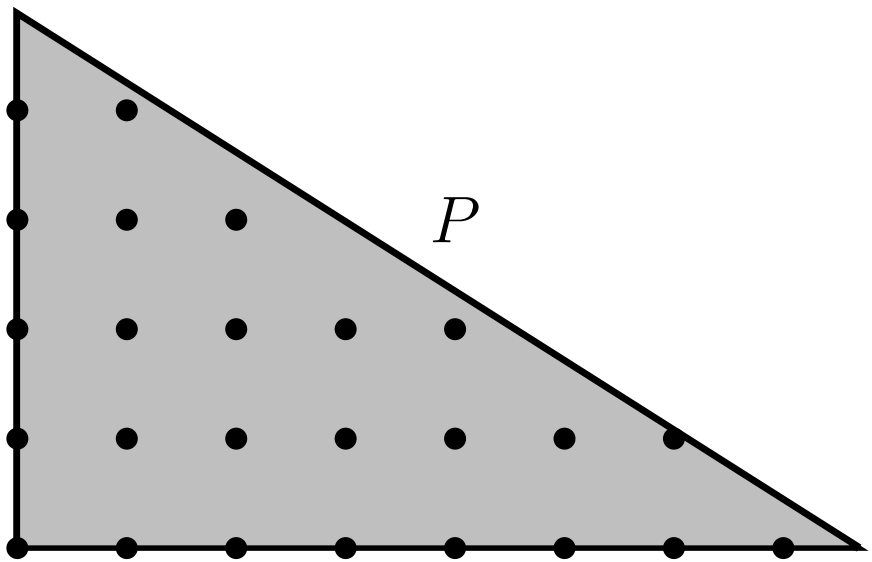
- Consider **parallelepiped** $\Pi \ni x$ with integral vertices
- Let y vertex of Π, in quadrant of x
- Let $z \in \Pi \cap \mathbb{Z}^d$ be mirrored point
- If $\lambda_x \geq 2 \Rightarrow$ redistribute weight
- At most 2^d points left inside Π



Covering a polytope with parallelepipeds

Lemma

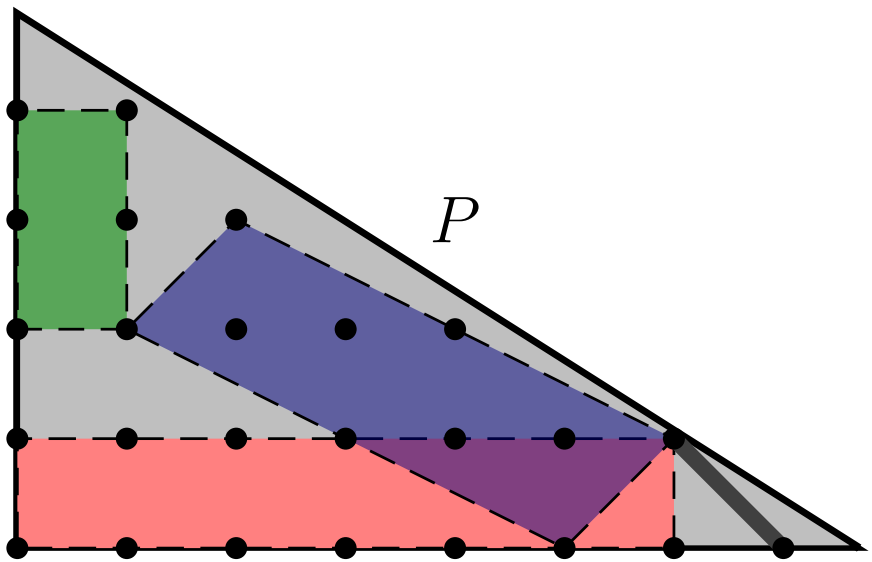
For fixed-dim $P \subseteq \mathbb{R}^d$, we can cover $P \cap \mathbb{Z}^d$ with **poly-many parallelepipeds** (with int. vertices and $\subseteq P$).



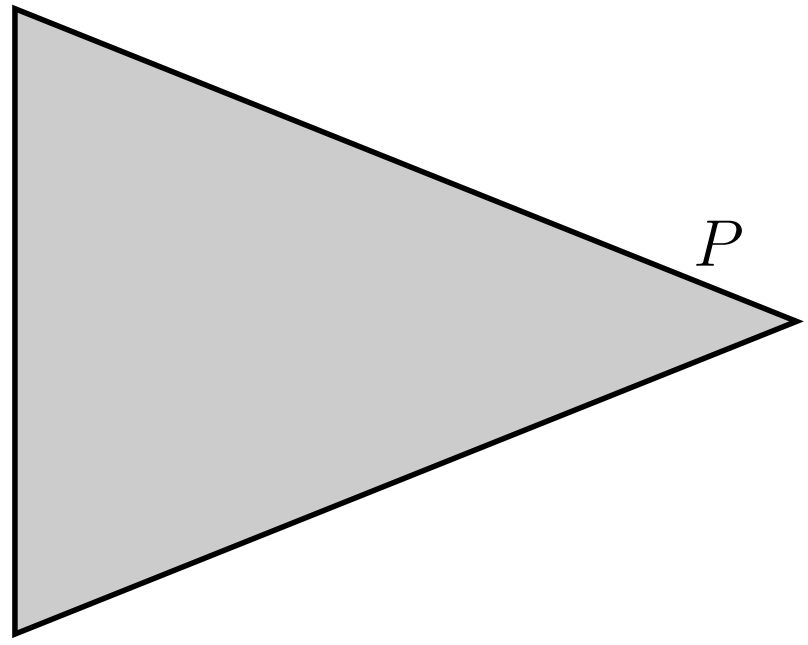
Covering a polytope with parallelepipeds

Lemma

For fixed-dim $P \subseteq \mathbb{R}^d$, we can cover $P \cap \mathbb{Z}^d$ with **poly-many parallelepipeds** (with int. vertices and $\subseteq P$).



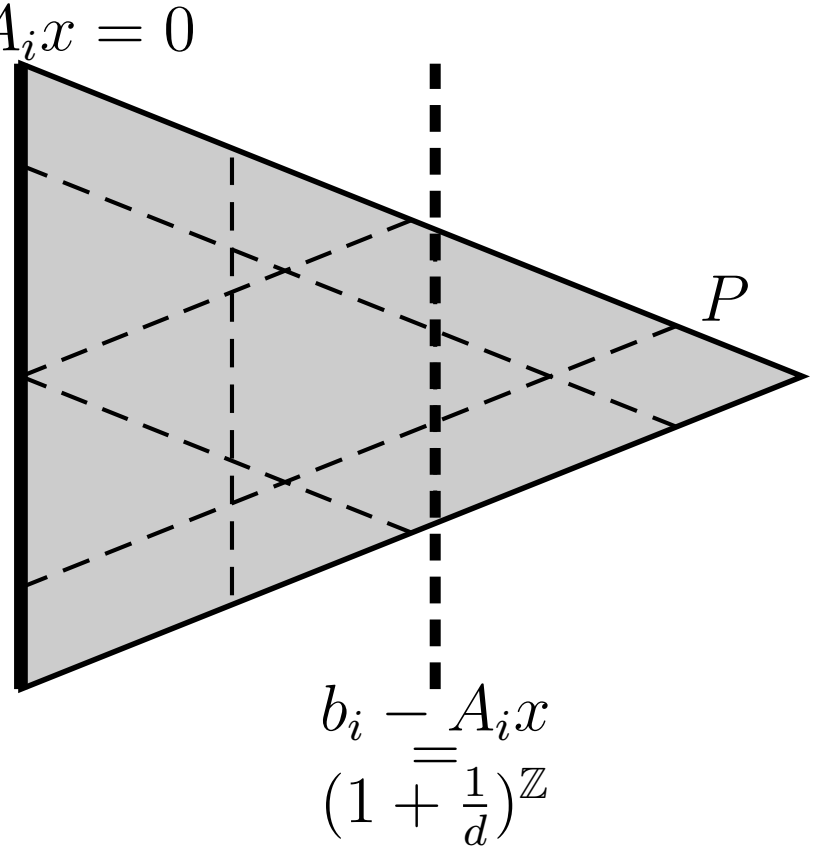
Covering a polytope w. parallelep. (2)



Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

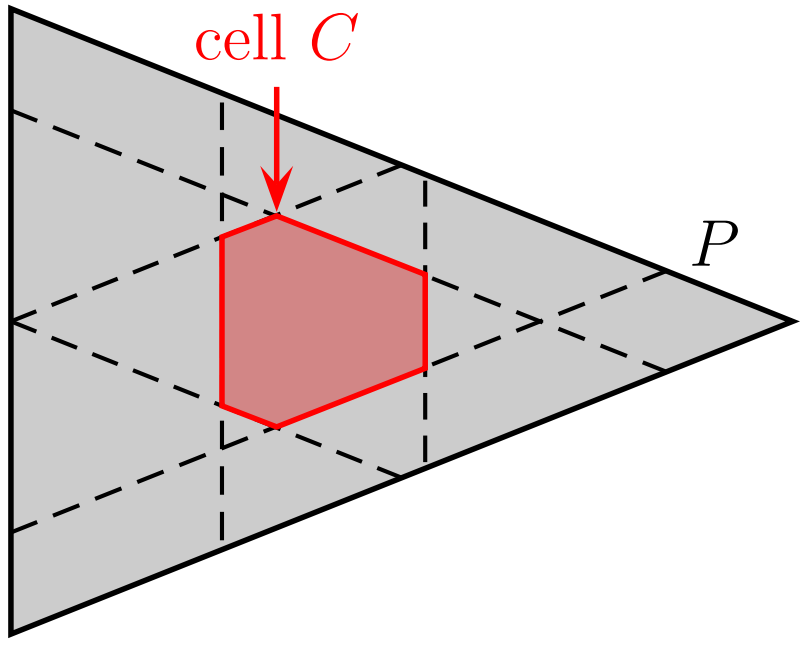
$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$



Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

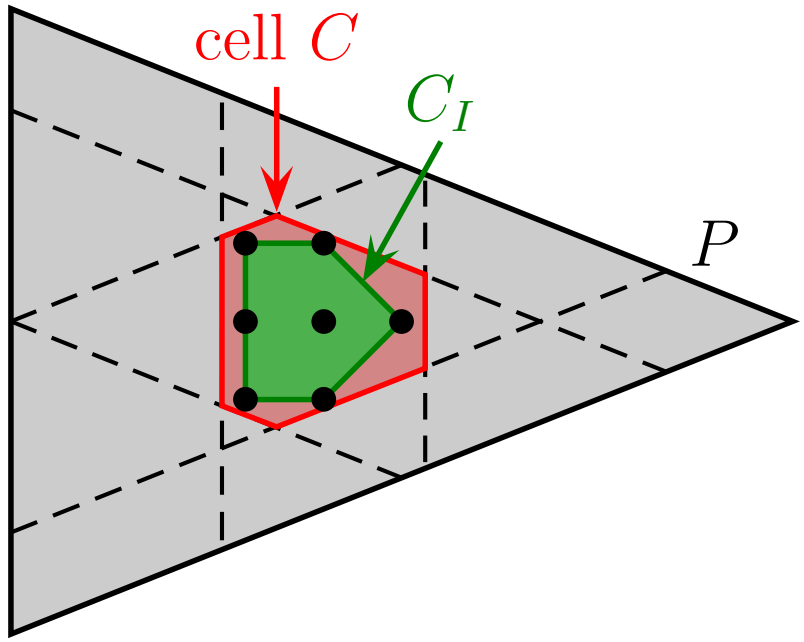


Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

- Consider int.hull C_I (poly many vertices)

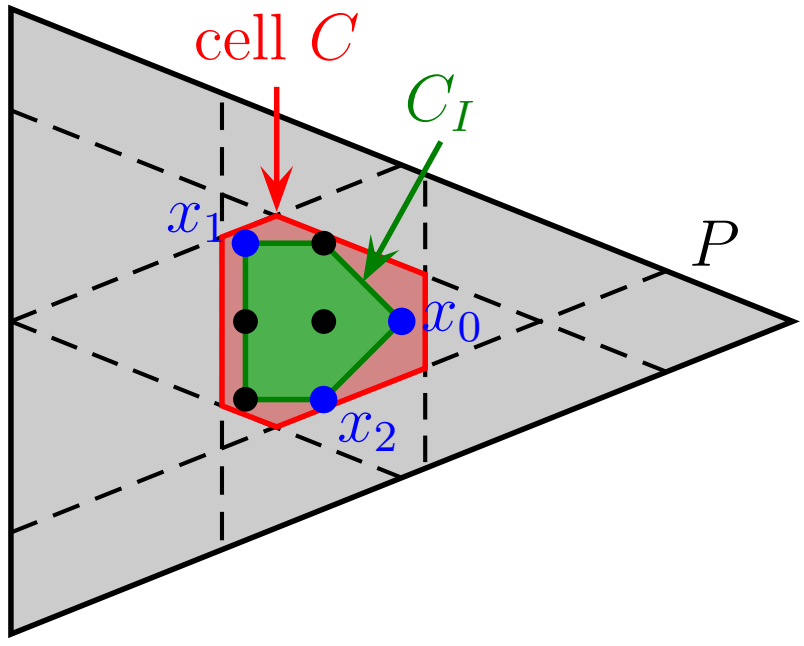


Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

- Consider int.hull C_I (poly many vertices)
- Extend any $d+1$ vertices of C_I to parallelepiped

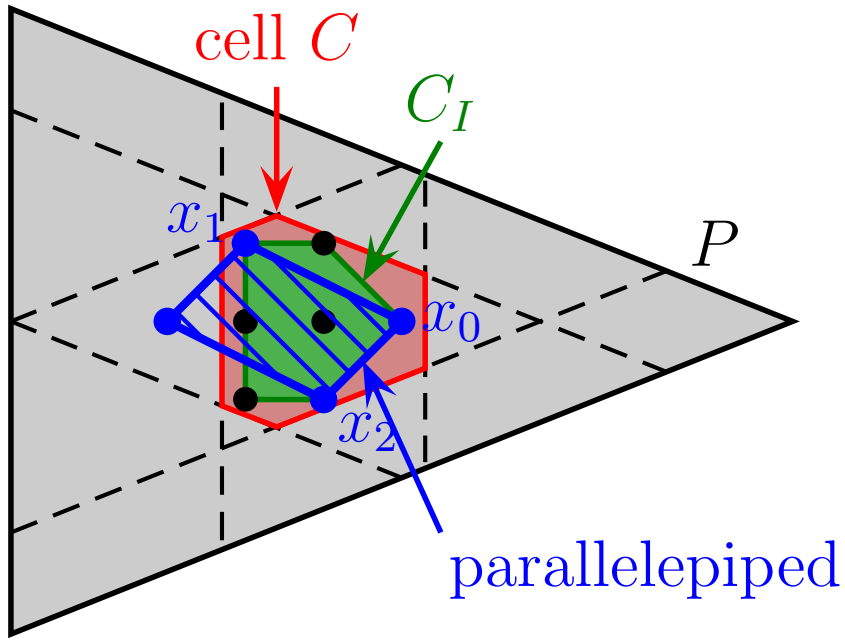


Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

- Consider int.hull C_I (poly many vertices)
- Extend any $d+1$ vertices of C_I to parallelepiped

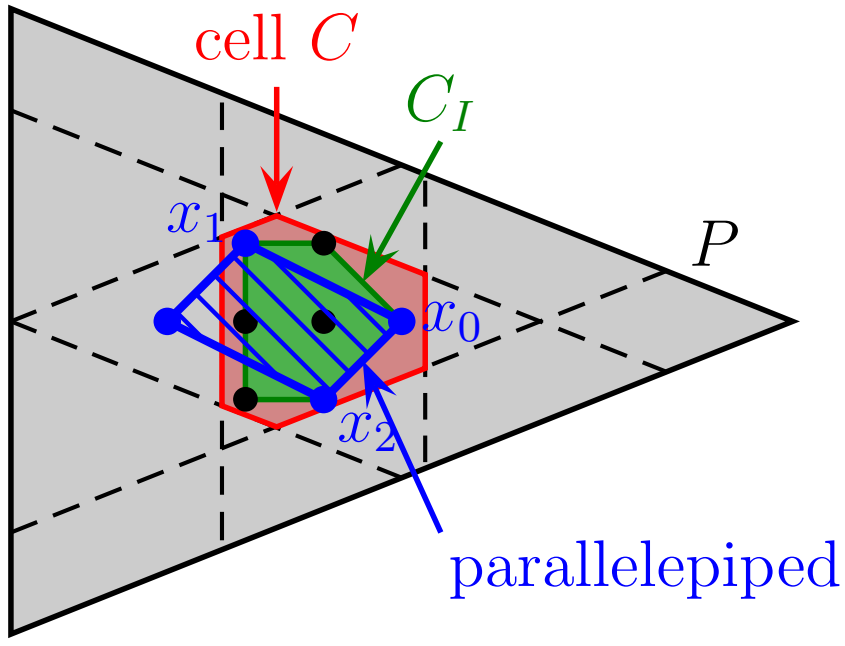


Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

- Consider int.hull C_I (poly many vertices)
- Extend any $d+1$ vertices of C_I to parallelepiped



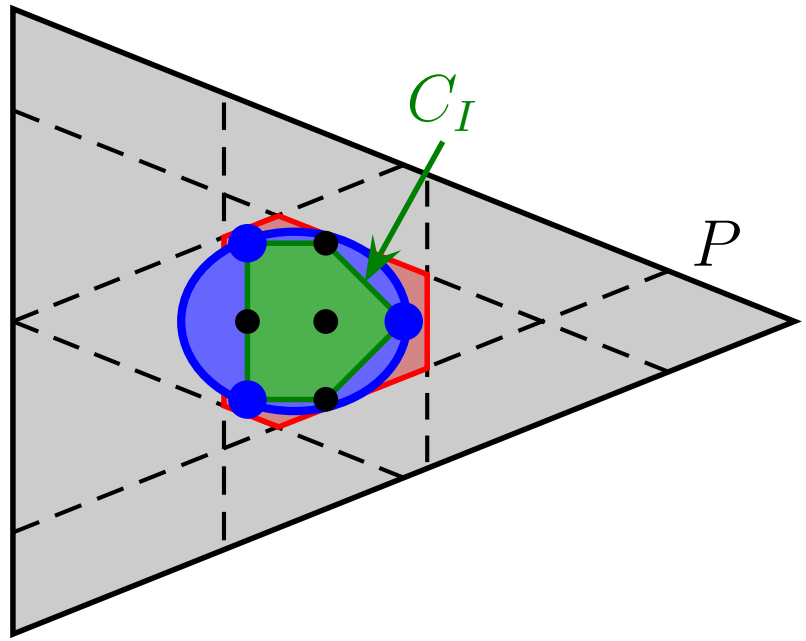
- Each cell covered with $d^{O(d^2)} m^{O(d^2)} (\log \Delta)^{O(d^2)}$ parallelepipeds

Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

- Consider int.hull C_I (poly many vertices)
- Extend any $d+1$ vertices of C_I to parallelepiped



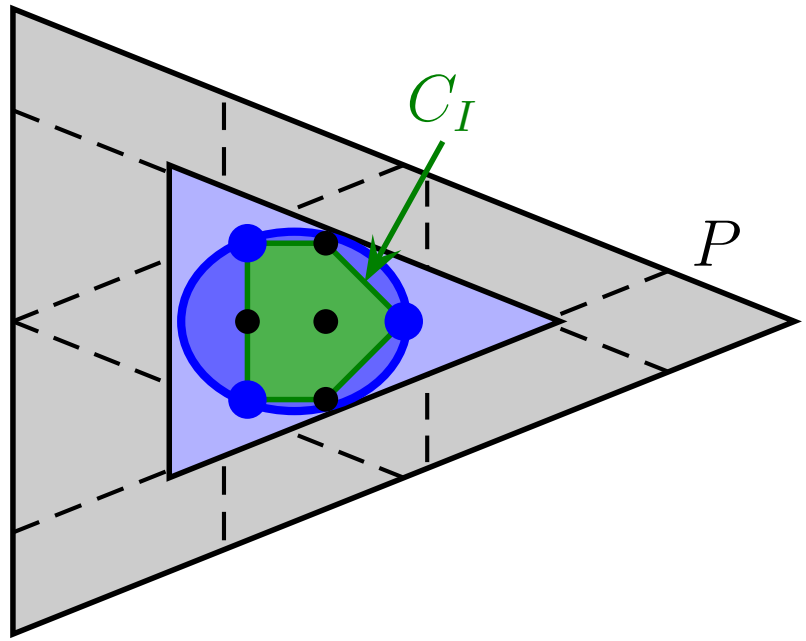
- Each cell covered with $d^{O(d^2)} m^{O(d^2)} (\log \Delta)^{O(d^2)}$ parallelepipeds
- **Improvement**: Approximate C_I by polytope with d^2 vertices using **John's ellipsoid**

Covering a polytope w. parallelep. (2)

- Split $P = \{x \mid Ax \leq b\}$ into poly many **cells**

$C = \{x \mid \alpha_{j(i)} \leq A_i x \leq \alpha_{j(i)+1}\}$

- Consider int.hull C_I (poly many vertices)
- Extend any $d+1$ vertices of C_I to parallelepiped



- Each cell covered with $d^{O(d^2)} m^{O(d^2)} (\log \Delta)^{O(d^2)}$ parallelepipeds
- **Improvement**: Approximate C_I by polytope with d^2 vertices using **John's ellipsoid**

The algorithm

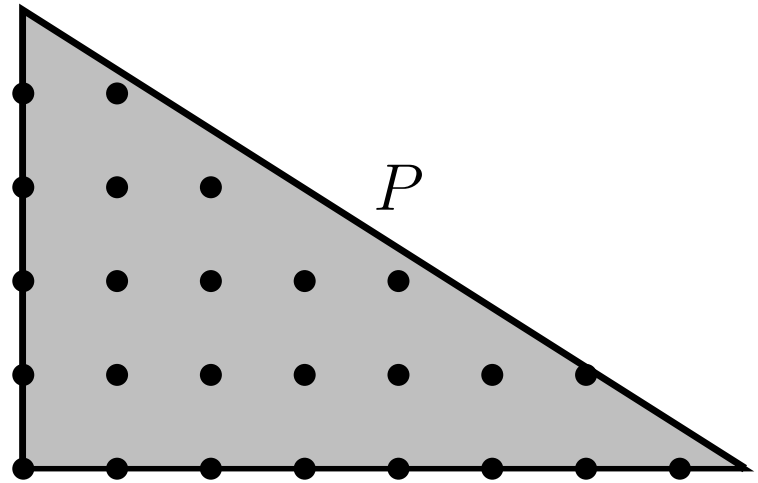
- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

Algorithm:

(1) Compute poly many parallelepipeds covering P

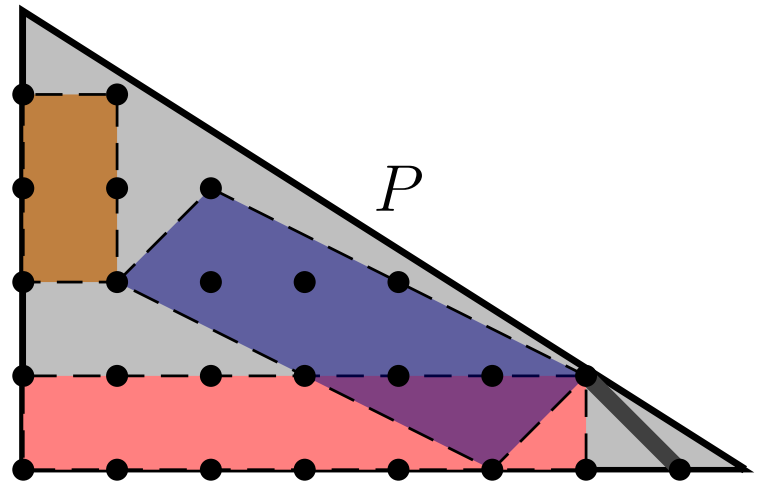


The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

Algorithm:

(1) Compute poly many parallelepipeds covering P



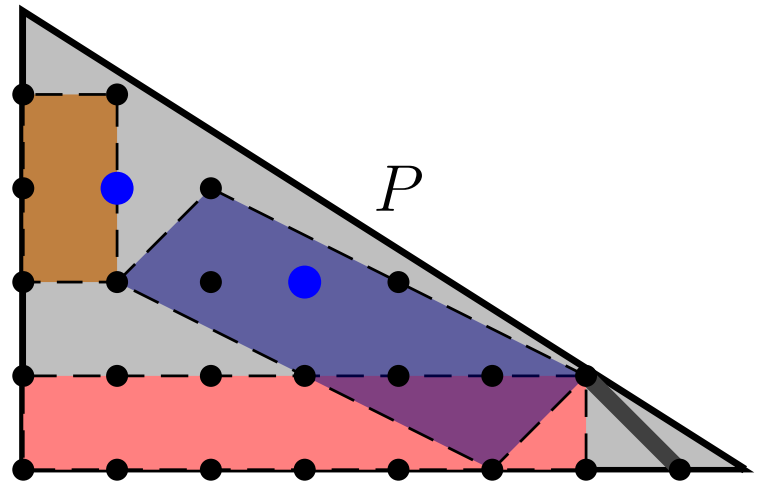
The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

Algorithm:

(1) Compute poly many parallelepipeds covering P

(2) Guess the 2^d parallelepipeds containing solution



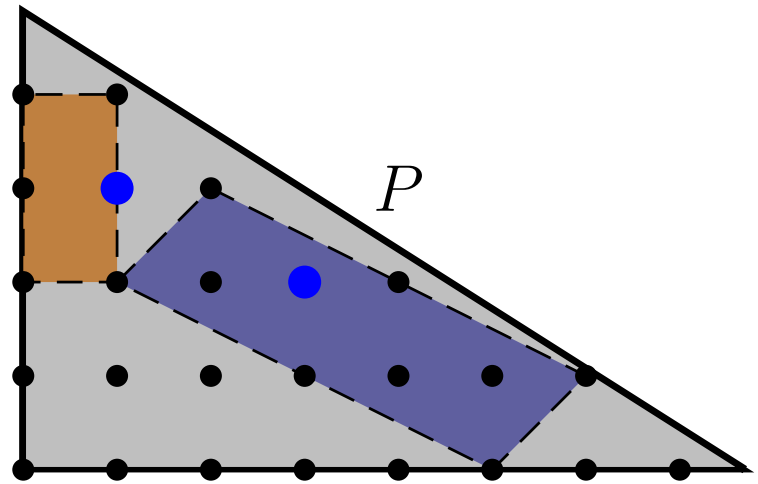
The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

Algorithm:

(1) Compute poly many parallelepipeds covering P

(2) Guess the 2^d parallelepipeds containing solution



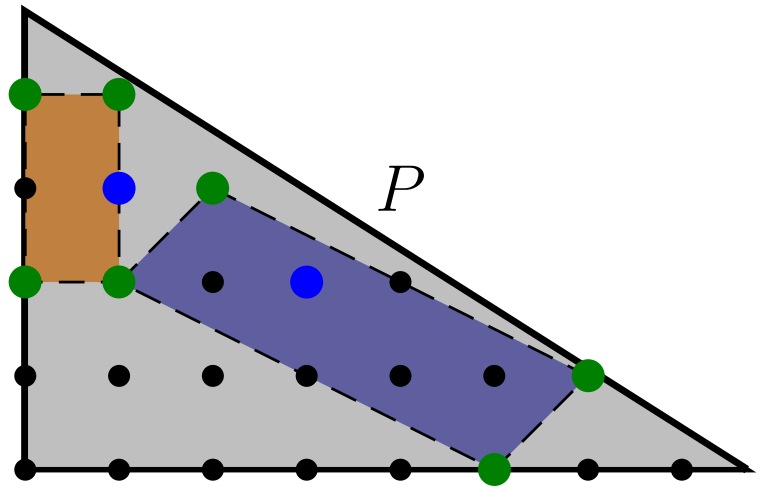
The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

Algorithm:

(1) Compute poly many parallelepipeds covering P

(2) Guess the 2^d parallelepipeds containing solution
$\rightarrow X :=$ vertices



The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

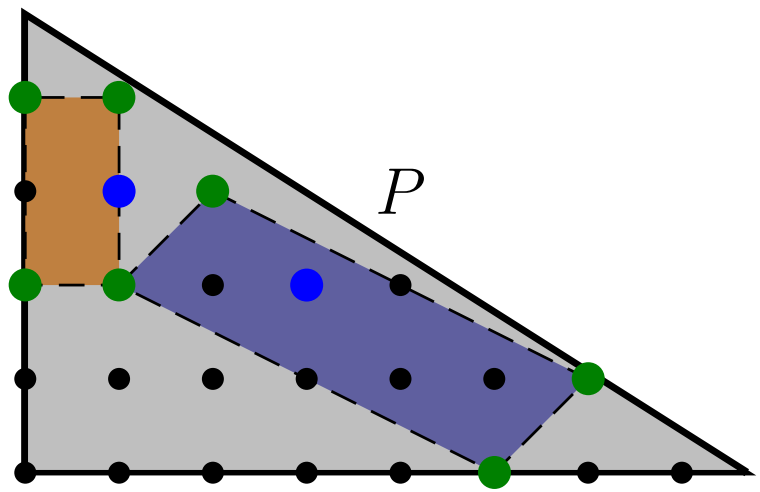
Algorithm:

(1) Compute poly many parallelepipeds covering P

(2) Guess the 2^d parallelepipeds containing solution
$\rightarrow X :=$ vertices

(3) Solve ILP with $2^{O(d)}$ variables

$$\sum_{x \in X} \lambda_x \cdot x + \sum_{\text{some } x \in P \cap \mathbb{Z}^d} 1 \cdot x \in Q$$



The algorithm

- **Input:** polytopes P, Q in ineq. description
- **Output:** Coefficients for int.cone$(P \cap \mathbb{Z}^d) \cap Q \neq \emptyset$

Algorithm:

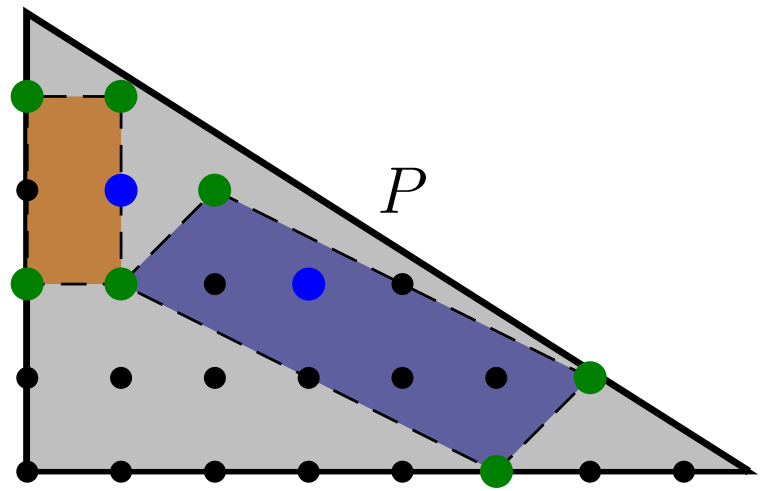
(1) Compute poly many parallelepipeds covering P

(2) Guess the 2^d parallelepipeds containing solution
$\rightarrow X :=$ vertices

(3) Solve ILP with $2^{O(d)}$ variables

$$\sum_{x \in X} \lambda_x \cdot x + \sum_{\text{some } x \in P \cap \mathbb{Z}^d} 1 \cdot x \in Q$$

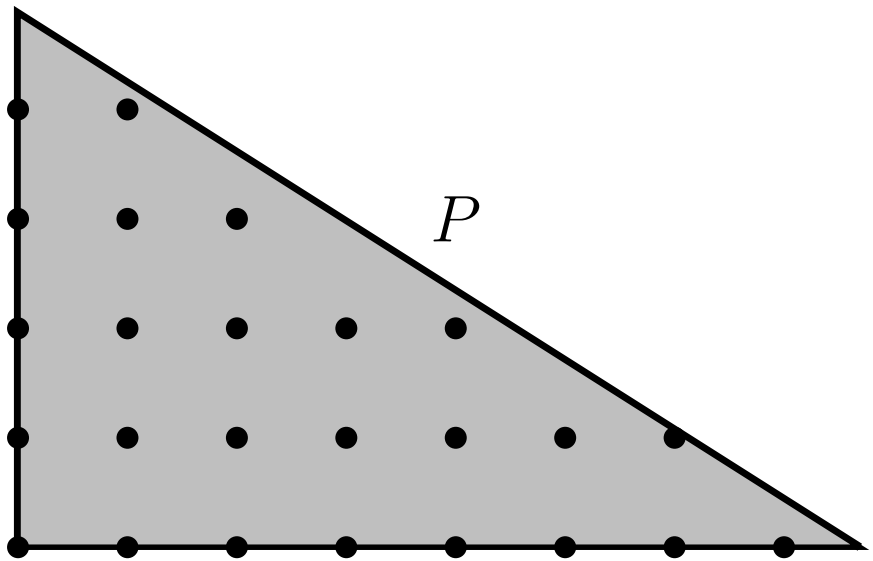
variables



A Structure Theorem

Structure Theorem

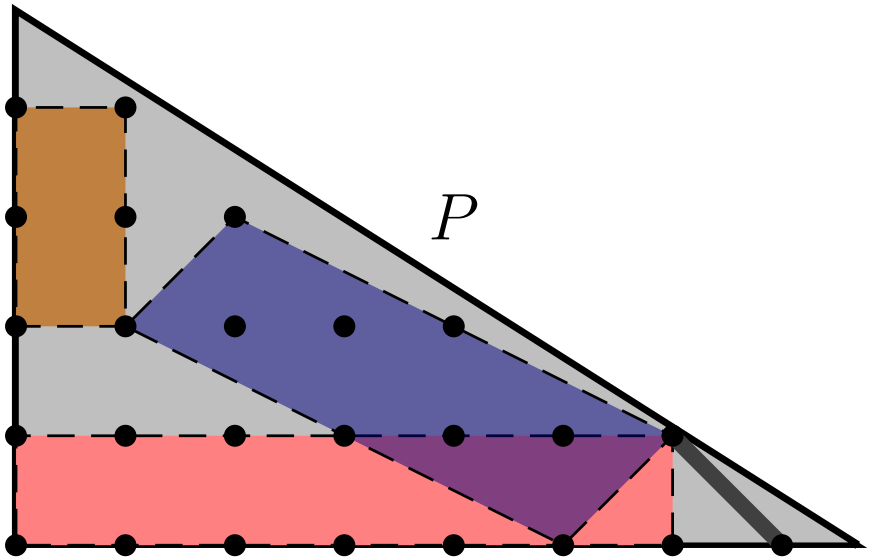
For polytope P



A Structure Theorem

Structure Theorem

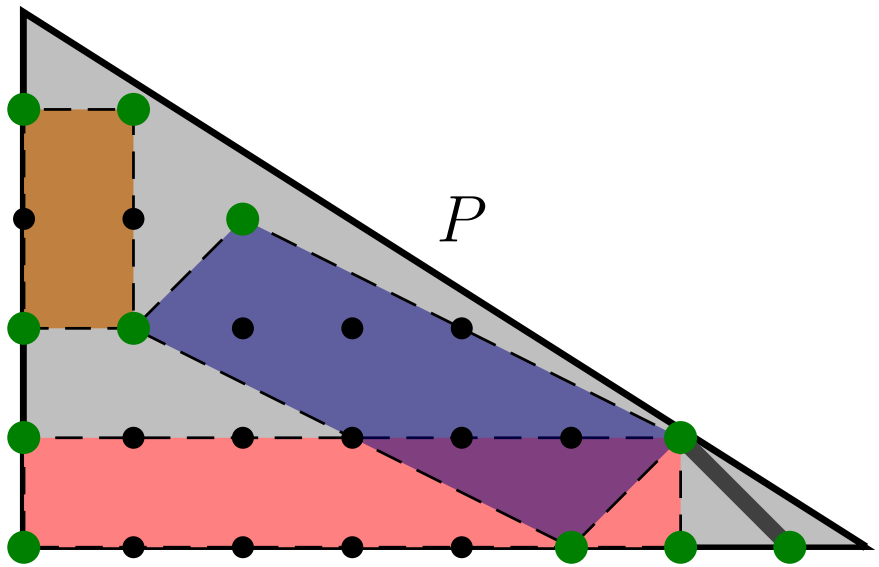
For polytope P



A Structure Theorem

Structure Theorem

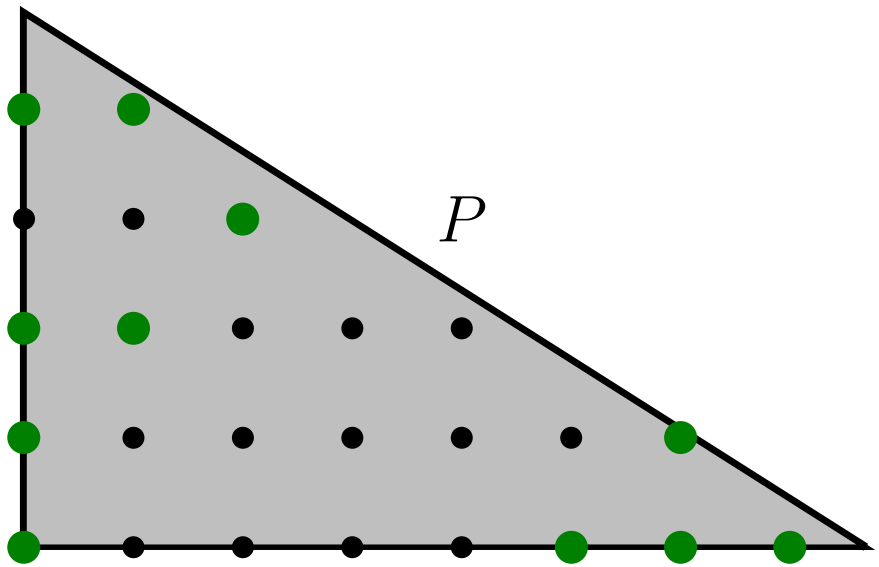
For polytope P, $\exists$ poly-time comp. set $\boldsymbol{X} \subseteq P \cap \mathbb{Z}^d$



A Structure Theorem

Structure Theorem

For polytope P, $\exists$ poly-time comp. set $\boldsymbol{X} \subseteq P \cap \mathbb{Z}^d$

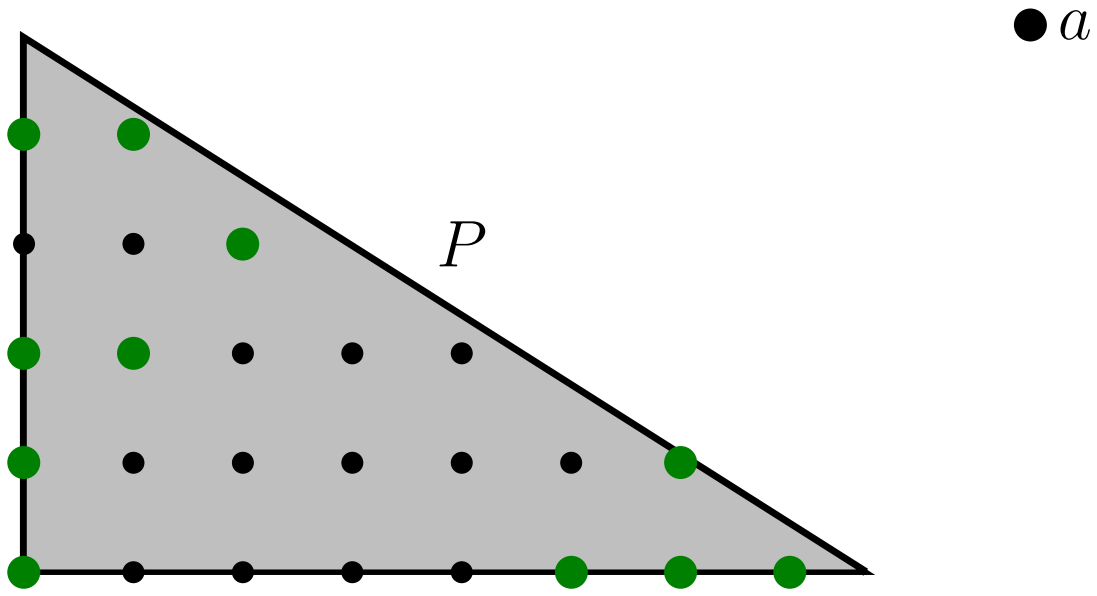


A Structure Theorem

Structure Theorem

For polytope P, $\exists$ poly-time comp. set $\boldsymbol{X} \subseteq P \cap \mathbb{Z}^d$ s.t. **for all** $a \in$ **int.cone**$(P \cap \mathbb{Z}^d)$ one can express

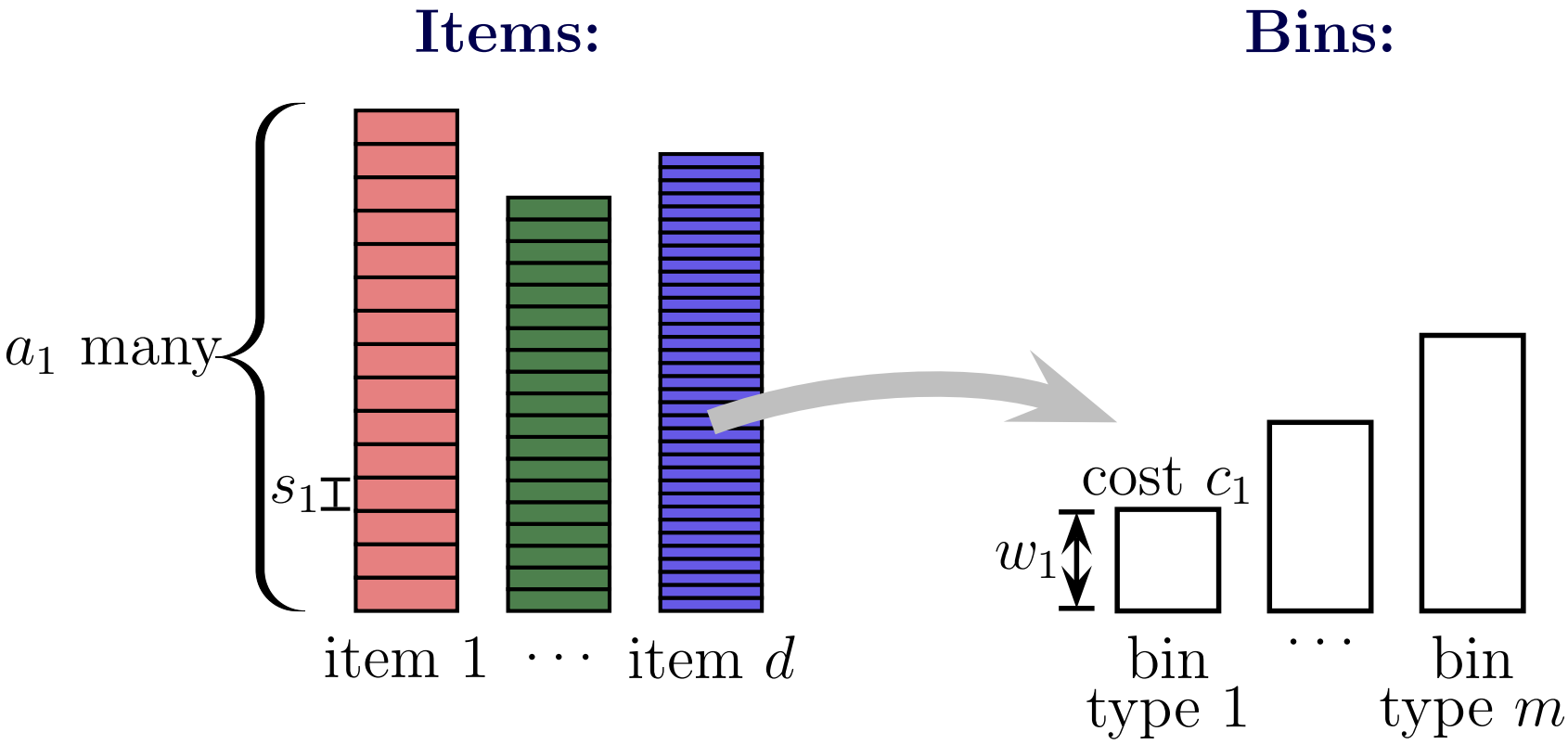
$$a = \text{int.cone}(2^{2d} \text{ points in } X) + \sum \text{of } 2^{2d} \text{ points in } P \cap \mathbb{Z}^d$$



More applications (1)

Theorem

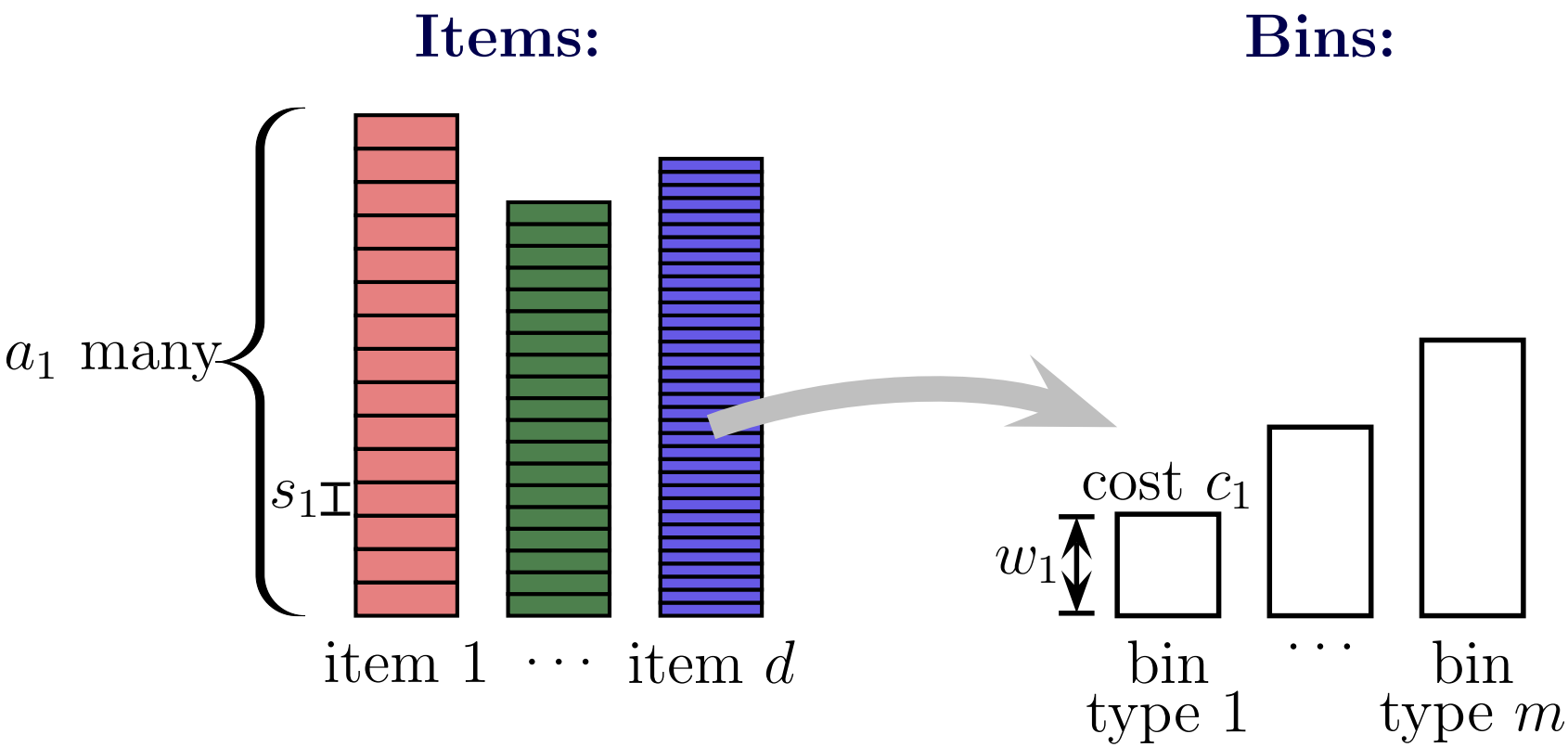
Cutting stock with $d = O(1)$ item types and $m = O(1)$ of bin types can be solved in **polytime**.



More applications (1)

Theorem

Cutting stock with $d = O(1)$ item types and $m = O(1)$ of bin types can be solved in **polytime**.



- $P = P_1 \times \ldots \times P_m$ with $P_j := \{ \binom{x}{c_j} \mid s^T x \leq w_j \}$

More applications (2)

High Multiplicity Scheduling:

- **Input:** d job types, each with release time, deadline, processing time, multiplicity
- **Goal:** Schedule jobs to minimize number of used machines

More applications (2)

High Multiplicity Scheduling:

- **Input:** d job types, each with release time, deadline, processing time, multiplicity
- **Goal:** Schedule jobs to minimize number of used machines

Theorem

High Multiplicity Scheduling can be solved in poly-time for fixed d.

The end

Thanks for your attention