An Algorithmic Theory of Integer Programming

Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein,
Martin Koutecký, Asaf Levin, Shmuel Onn

MIP 2019, MIT

TECHNION
Israel Institute
of Technology

CHARLES UNIVERSITY
Faculty of mathematics
and physics

Outline

Outline

My goal is to sell you this paper:

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, Shmuel Onn

Outline

My goal is to sell you this paper:

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, Shmuel Onn

Talk Outline:

1. Integer Programming: Structural Parameterizations
2. The Theory: Iterative Augmentation
3. The Extras: Proximity, Scaling, Reducibility, Near-linear / Strongly Polynomial Algorithms, Lower Bounds, etc.
4. Outlook

My goal is to sell you this paper:

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, Shmuel Onn

Talk Outline:

1. Integer Programming: Structural Parameterizations
2. The Theory: Iterative Augmentation
3. The Extras: Proximity, Scaling, Reducibility, Near-linear / Strongly Polynomial Algorithms, Lower Bounds, etc.
4. Outlook

Outline

My goal is to sell you this paper:

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, Shmuel Onn

Talk Outline:

1. Integer Programming: Structural Parameterizations
2. The Theory: Iterative Augmentation
3. The Extras: Proximity, Scaling, Reducibility, Near-linear / Strongly Polynomial Algorithms, Lower Bounds, etc.
4. Outlook

Outline

My goal is to sell you this paper:

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, Shmuel Onn

Talk Outline:

1. Integer Programming: Structural Parameterizations
2. The Theory: Iterative Augmentation
3. The Extras: Proximity, Scaling, Reducibility, Near-linear / Strongly Polynomial Algorithms, Lower Bounds, etc.
4. Outlook

Integer Programming: Structural Parameterizations

Variable Dimension: Unifying Theory

👁 **Real world is high-dimensional!**

Brief history of variable dimension IP:

- 1960's: Total Unimodularity (paths, matchings, flows) [Hoffman, Kruskal]
- 1980's: ILPs with few rows (generalized knapsack) [Papadimitriou; Eisenbrand, Weismantel; Jansen, Rohwedder]
- 2010–: Iterative methods for block structured programs [Aschenbrenner, Chen, De Loera, Hemmecke, Köppe, Lee, Marx, Onn, Romanchuk, Schulz, Weismantel]
- 2015–: Tree-structured ILPs [Ganian, Jansen, Kratsch, Ordyniak, Ramanujan]

Variable Dimension: Unifying Theory

👁 **Real world is high-dimensional!**

Brief history of variable dimension IP:

- 1980's: ILPs with few rows (generalized knapsack)
  [Papadimitriou; Eisenbrand, Weismantel; Jansen, Rohwedder]
- 2010–: Iterative methods for block structured programs [Aschenbrenner, Chen, De Loera, Hemmecke, Köppe, Lee, Marx, Onn, Romanchuk, Schulz, Weismantel]
- 2015–: Tree-structured ILPs
  [Ganian, Jansen, Kratsch, Ordyniak, Ramanujan]

No strongly polynomial algorithms for these classes (and few overall: TU, bimodular, binet).

Variable Dimension: Unifying Theory

Real world is high-dimensional!

Brief history of variable dimension IP:

- 1980's: ILPs with few rows (generalized knapsack) [proximity, DP]
- 2010–: Iterative methods for block structured programs [augmentation, Ramsey, Algebra, DP]
- 2015–: Tree-structured ILPs [Lenstra, treewidth]

No strongly polynomial algorithms for these classes (and few overall: TU, bimodular, binet). Seemingly disconnected classes, different methods.

Variable Dimension: Unifying Theory

⇓

👁 **Real world is high-dimensional!**

Brief history of variable dimension IP:

- 1980's: ILPs with few rows (generalized knapsack)
- 2010–: Iterative methods for block structured programs
- 2015–: Tree-structured ILPs

No strongly polynomial algorithms for these classes (and few overall: TU, bimodular, binet). Seemingly disconnected classes, different methods.

Our result: improves, unifies, simplifies, makes strongly-poly *all* of these results!

Structural Parameterizations: The Graphs of A

$$\min f(\mathbf{x}) \; : \; A\mathbf{x} = \mathbf{b}, \; \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \; \mathbf{x} \in \mathbb{Z}^n \qquad \text{(IP)}$$

Structural Parameterizations: The Graphs of A

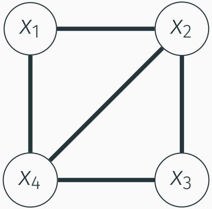
$$\min f(\mathbf{x}) \ : \ A\mathbf{x} = \mathbf{b},\ \mathbf{l} \leq \mathbf{x} \leq \mathbf{u},\ \mathbf{x} \in \mathbb{Z}^n \qquad \text{(IP)}$$

$$
\begin{aligned}
x_1 + 2x_2 + x_4 &= 11 && (C_1)\\
3x_3 - 2x_4 &= 6 && (C_2)\\
x_2 + 2x_3 &= 2 && (C_3)\\
-x_1 + 3x_4 &= 2 && (C_4)\\
0 \leq x_1, x_2, x_3, x_4 &\leq 5 && \text{(box)}
\end{aligned}
$$

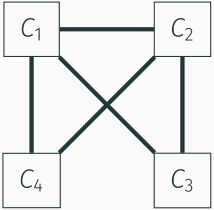
Structural Parameterizations: The Graphs of A

$$\min f(\mathbf{x}) \;:\; A\mathbf{x} = \mathbf{b},\ \mathbf{l} \le \mathbf{x} \le \mathbf{u},\ \mathbf{x} \in \mathbb{Z}^n \qquad \text{(IP)}$$

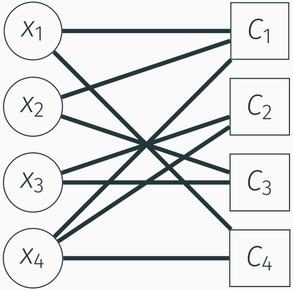
$$\begin{aligned}
x_1 + 2x_2 + x_4 &= 11 && (C_1)\\
3x_3 - 2x_4 &= 6 && (C_2)\\
x_2 + 2x_3 &= 2 && (C_3)\\
-x_1 + 3x_4 &= 2 && (C_4)\\
0 \le x_1, x_2, x_3, x_4 &\le 5 && (\text{box})
\end{aligned}$$



Primal: $G_P(A)$



Dual: $G_D(A)$



Incidence: $G_I(A)$

Structural Parameterizations: Small Graphs $\approx$ Classics

Lenstra '83 $\Rightarrow$ ILP solvable in time $f_P(|G_P|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

Structural Parameterizations: Small Graphs $\approx$ Classics

Lenstra '83 $\Rightarrow$ ILP solvable in time $f_P(|G_P|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

Parameterized complexity perspective:

- Runtime $f(\alpha) \cdot \text{poly}(\beta)$ with
  parameter $\alpha = |G_P| = n$ and
  input $\beta = \langle A, \mathbf{w}, \mathbf{b} \rangle$

$f(\alpha)\,\text{poly}(\beta)$	clearly better than	$\beta^{f(\alpha)}$
FPT (fixed-parameter tractable)		XP

IP has many natural parameters: dimension n, #rows m, largest coefficient $\|A\|_\infty$, etc.

Structural Parameterizations: Small Graphs $\approx$ Classics

Lenstra '83 $\Rightarrow$ ILP solvable in time $f_P(|G_P|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

because $|G_P| < |G_I|$ also in $f_P(|G_I|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

Structural Parameterizations: Small Graphs $\approx$ Classics

Lenstra '83 $\Rightarrow$ ILP solvable in time $f_P(|G_P|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

because $|G_P| < |G_I|$ also in $f_P(|G_I|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

Papadimitriou '81 $\Rightarrow$ ILP solvable in time $f_D(|G_D|, \|A\|_\infty) \cdot n \cdot \langle \mathbf{b}, \mathbf{w} \rangle$

can't parameterize only by $|G_D|$ or $\|A\|_\infty$.

Structural Parameterizations: Small Graphs $\approx$ Classics

Lenstra '83 $\Rightarrow$ ILP solvable in time $f_P(|G_P|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

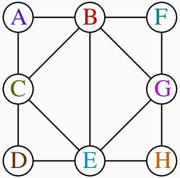
because $|G_P| < |G_I|$ also in $f_P(|G_I|) \cdot \langle A, \mathbf{b}, \mathbf{w} \rangle$

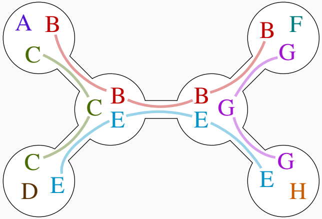
Papadimitriou '81 $\Rightarrow$ ILP solvable in time $f_D(|G_D|, \|A\|_\infty) \cdot n \cdot \langle \mathbf{b}, \mathbf{w} \rangle$

can't parameterize only by $|G_D|$ or $\|A\|_\infty$.

Graph size = too strict a parameter. What else?

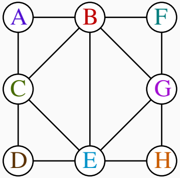
Structural Parameterizations: Treewidth

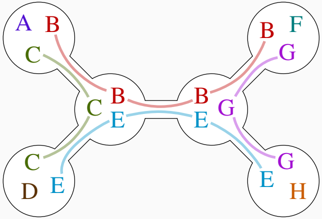




+ a popular and successful parameter

Structural Parameterizations: Treewidth





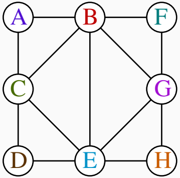
\+ a popular and successful parameter

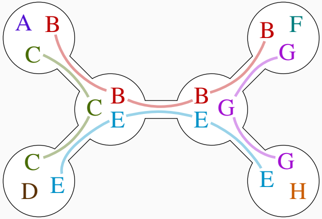
– not for IP

NP-hard even for $\text{tw}_P, \text{tw}_D, \text{tw}_I \leq 3$, $\|A\|_\infty = 2$

($\text{tw}_P(A) = \text{tw}(G_P(A))$, $\text{tw}_D(A) = \text{tw}(G_D(A))$, $\text{tw}_I(A) = \text{tw}(G_I(A))$.)

Structural Parameterizations: Treewidth





+ a popular and successful parameter

– not for IP

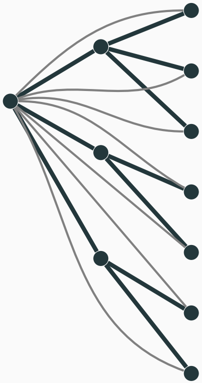
NP-hard even for $\mathrm{tw}_P, \mathrm{tw}_D, \mathrm{tw}_I \leq 3$, $\|A\|_\infty = 2$

($\mathrm{tw}_P(A) = \mathrm{tw}(G_P(A))$, $\mathrm{tw}_D(A) = \mathrm{tw}(G_D(A))$, $\mathrm{tw}_I(A) = \mathrm{tw}(G_I(A))$.)

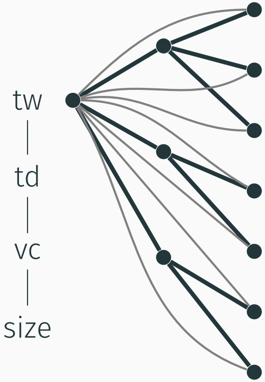
A more restrictive parameter?

Structural Parameterizations: Treedepth

tw
|
td
|
vc
|
size



$$\mathrm{td}(G) = \begin{cases} 1 & \text{if } |V(G)| = 1, \\ 1 + \min_{v \in V(G)} \mathrm{td}(G - v) & \text{if connected,} \\ \max_{G_i \text{ component}} \mathrm{td}(G_i) & \text{if disconnected} \end{cases}$$



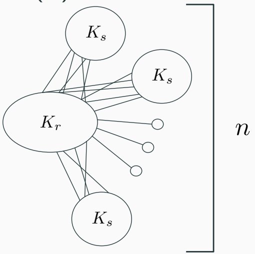
$$\mathrm{td}(G) = \begin{cases} 1 & \text{if } |V(G)| = 1, \\ 1 + \min_{v \in V(G)} \mathrm{td}(G - v) & \text{if connected,} \\ \max_{G_i \text{ component}} \mathrm{td}(G_i) & \text{if disconnected} \end{cases}$$

Example: n-fold IP matrix

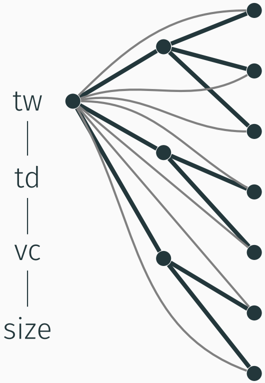
$$A = \underbrace{\begin{pmatrix} A_1 & A_1 & \cdots & A_1 \\ A_2 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_2 \end{pmatrix}}_{n}$$

$$\begin{array}{c|c} \overbrace{A_1}^{t} & r \\ A_2 & s \end{array}$$

$G_D(A)$

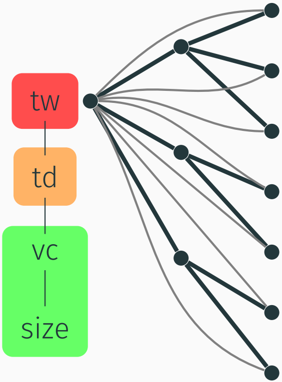


Structural Parameterizations: Treedepth



$$\mathrm{td}(G) = \begin{cases} 1 & \text{if } |V(G)| = 1, \\ 1 + \min_{v \in V(G)} \mathrm{td}(G - v) & \text{if connected}, \\ \max_{G_i \text{ component}} \mathrm{td}(G_i) & \text{if disconnected} \end{cases}$$

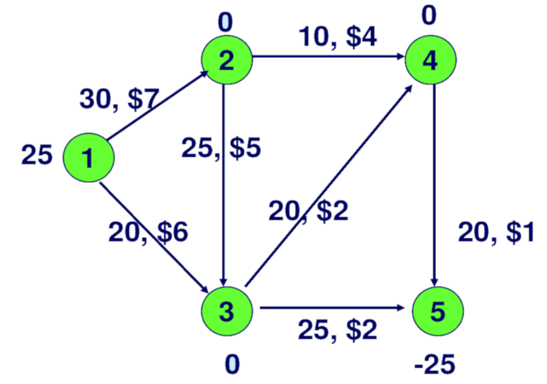
- **Main result: IP solvable in time**
  $g(\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}, \|A\|_\infty) \cdot \mathrm{poly}(n)$
  $\Rightarrow$ FPT par. by $\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}$ and $\|A\|_\infty$



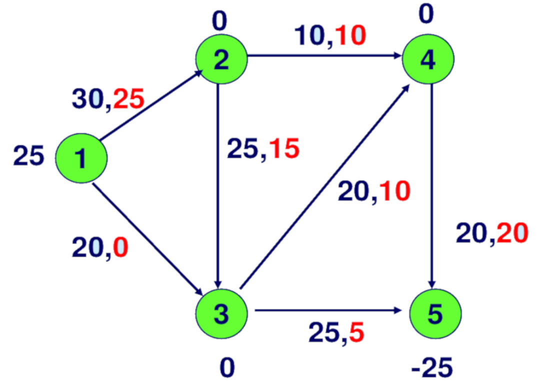
$$\mathrm{td}(G) = \begin{cases} 1 & \text{if } |V(G)| = 1, \\ 1 + \min_{v \in V(G)} \mathrm{td}(G - v) & \text{if connected,} \\ \max_{G_i \text{ component}} \mathrm{td}(G_i) & \text{if disconnected} \end{cases}$$

- **Main result: IP solvable in time** $g(\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}, \|A\|_\infty) \cdot \mathrm{poly}(n)$
  $\Rightarrow$ FPT par. by $\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}$ and $\|A\|_\infty$
- ILP NP-h for $\mathrm{td}_I = 5$ and $\|A\|_\infty = 1$ [Eiben et al. '19]

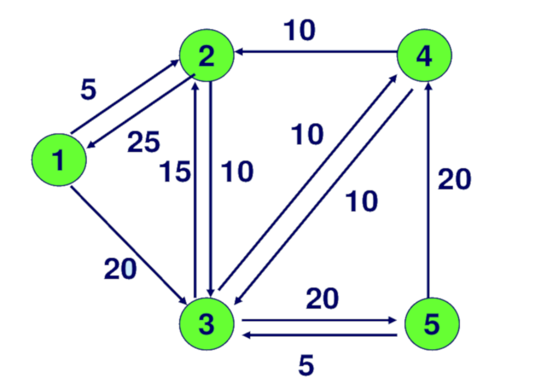
The Theory: Iterative Augmentation



Min-cost flow

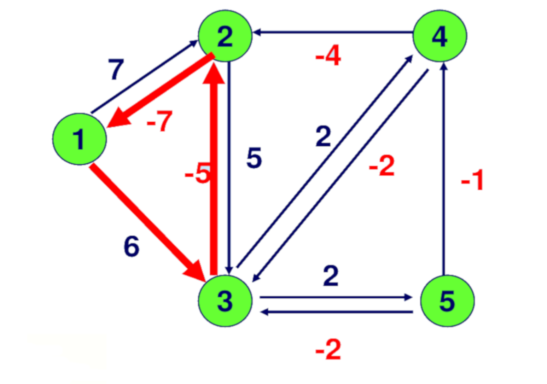


Min-cost flow



Min-cost flow

Iterative Augmentation



Min-cost flow

a step $\equiv$ a cycle C

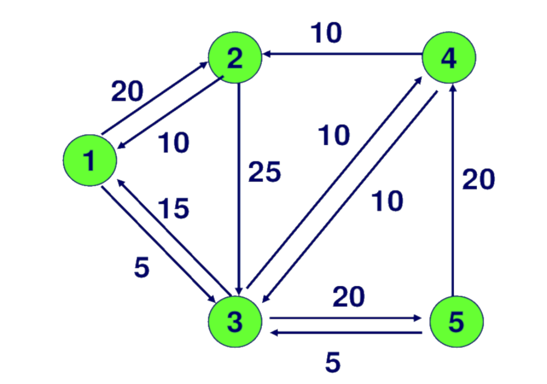
(because circulations decompose into cycles)

C *feasible* if enough capacity (fits res. net)

C *augmenting* if negative

flow cost *minimal* if $\nexists$ negative cycle

Iterative Augmentation



Min-cost flow

a step $\equiv$ a cycle C

(because circulations decompose into cycles)

C *feasible* if enough capacity (fits res. net)

C *augmenting* if negative

flow cost *minimal* if $\nexists$ negative cycle

Iterative Augmentation

$$\min f(\mathbf{x}) \;:\; A\mathbf{x} = \mathbf{b},\, \mathbf{l} \le \mathbf{x} \le \mathbf{u},\, \mathbf{x} \in \mathbb{Z}^n$$

Integer Programming

$\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A) = \{\mathbf{g} \in \mathbb{Z}^n \mid A\mathbf{g} = \mathbf{0}\}$
$(A\mathbf{x} = \mathbf{b} \implies A(\mathbf{x} + \mathbf{g}) = \mathbf{b})$

$\mathbf{g}$ *feasible* if $\mathbf{l} \le \mathbf{x} + \mathbf{g} \le \mathbf{u}$

$\mathbf{g}$ *augmenting* if $f(\mathbf{x} + \mathbf{g}) < f(\mathbf{x})$

$\mathbf{x}$ *optimal* if $\nexists$ augmenting $\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A)$

Min-cost flow

a step $\equiv$ a cycle C
(because circulations decompose into cycles)

C *feasible* if enough capacity (fits res. net)

C *augmenting* if negative

flow cost *minimal* if $\nexists$ negative cycle

$$\min f(\mathbf{x}) \;:\; A\mathbf{x} = \mathbf{b},\ \mathbf{l} \le \mathbf{x} \le \mathbf{u},\ \mathbf{x} \in \mathbb{Z}^n$$

Integer Programming

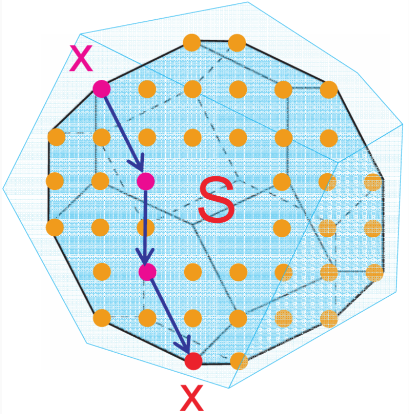
$\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A) = \{\mathbf{g} \in \mathbb{Z}^n \mid A\mathbf{g} = \mathbf{0}\}$

$(A\mathbf{x} = \mathbf{b} \implies A(\mathbf{x} + \mathbf{g}) = \mathbf{b})$

$\mathbf{g}$ *feasible* if $\mathbf{l} \le \mathbf{x} + \mathbf{g} \le \mathbf{u}$

$\mathbf{g}$ *augmenting* if $f(\mathbf{x} + \mathbf{g}) < f(\mathbf{x})$

$\mathbf{x}$ *optimal* if $\nexists$ augmenting $\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A)$



$$\min f(\mathbf{x}) \;:\; A\mathbf{x} = \mathbf{b},\ \mathbf{l} \le \mathbf{x} \le \mathbf{u},\ \mathbf{x} \in \mathbb{Z}^n$$

Integer Programming

$\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A) = \{\mathbf{g} \in \mathbb{Z}^n \mid A\mathbf{g} = \mathbf{0}\}$

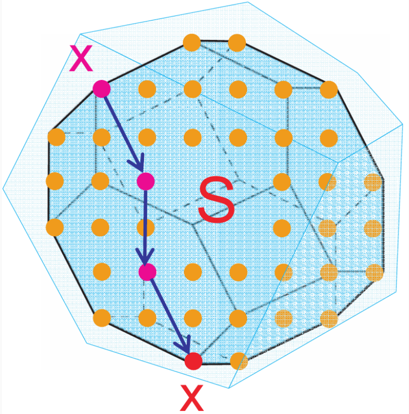
$(A\mathbf{x} = \mathbf{b} \implies A(\mathbf{x} + \mathbf{g}) = \mathbf{b})$

$\mathbf{g}$ *feasible* if $\mathbf{l} \le \mathbf{x} + \mathbf{g} \le \mathbf{u}$

$\mathbf{g}$ *augmenting* if $f(\mathbf{x} + \mathbf{g}) < f(\mathbf{x})$

$\mathbf{x}$ *optimal* if $\nexists$ augmenting $\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A)$

BUT $\mathrm{Ker}_{\mathbb{Z}}(A)$ too big and wild...



$$\min f(\mathbf{x}) \;:\; A\mathbf{x} = \mathbf{b},\ \mathbf{l} \le \mathbf{x} \le \mathbf{u},\ \mathbf{x} \in \mathbb{Z}^n$$

Integer Programming

$\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A) = \{\mathbf{g} \in \mathbb{Z}^n \mid A\mathbf{g} = \mathbf{0}\}$
$(A\mathbf{x} = \mathbf{b} \implies A(\mathbf{x} + \mathbf{g}) = \mathbf{b})$
$\mathbf{g}$ *feasible* if $\mathbf{l} \le \mathbf{x} + \mathbf{g} \le \mathbf{u}$
$\mathbf{g}$ *augmenting* if $f(\mathbf{x} + \mathbf{g}) < f(\mathbf{x})$
$\mathbf{x}$ *optimal* if $\nexists$ augmenting $\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A)$

BUT $\mathrm{Ker}_{\mathbb{Z}}(A)$ too big and wild...

Goal: Find $\mathcal{T} \subseteq \mathrm{Ker}_{\mathbb{Z}}(A)$, s.t.

1. $\mathbf{x}$ not opt then $\exists$ augmenting $\mathbf{g} \in \mathcal{T}$
2. good convergence for repeatedly adding "good" $\mathbf{g} \in \mathcal{T}$,
3. algorithmically tame (big is OK)

$$\min f(\mathbf{x}) \ : \ A\mathbf{x} = \mathbf{b},\ \mathbf{l} \leq \mathbf{x} \leq \mathbf{u},\ \mathbf{x} \in \mathbb{Z}^n$$

Integer Programming

$\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A) = \{\mathbf{g} \in \mathbb{Z}^n \mid A\mathbf{g} = \mathbf{0}\}$
$(A\mathbf{x} = \mathbf{b} \implies A(\mathbf{x} + \mathbf{g}) = \mathbf{b})$
$\mathbf{g}$ *feasible* if $\mathbf{l} \leq \mathbf{x} + \mathbf{g} \leq \mathbf{u}$
$\mathbf{g}$ *augmenting* if $f(\mathbf{x} + \mathbf{g}) < f(\mathbf{x})$
$\mathbf{x}$ *optimal* if $\nexists$ augmenting $\mathbf{g} \in \mathrm{Ker}_{\mathbb{Z}}(A)$

BUT $\mathrm{Ker}_{\mathbb{Z}}(A)$ too big and wild...

Goal: Find $\mathcal{T} \subseteq \mathrm{Ker}_{\mathbb{Z}}(A)$, s.t.

1. $\mathbf{x}$ not opt then $\exists$ augmenting $\mathbf{g} \in \mathcal{T}$
2. good convergence for repeatedly adding "good" $\mathbf{g} \in \mathcal{T}$,
3. algorithmically tame (big is OK)

Answer: ($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge\ |x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x} \text{ is } \sqsubseteq\text{-minimal}\}$

Iterative Augmentation (Sec 2.1)

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge |x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x}$ is $\sqsubseteq$ -minimal$\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge$ $|x_i| \le |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x} \text{ is } \sqsubseteq\text{-minimal}\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *Graver-best step* for $\mathbf{x}$ if as good as $\lambda\mathbf{g}$ for any $\lambda \in \mathbb{N}$, $\mathbf{g} \in \mathcal{G}(A)$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *halfling* if half as good as Graver-best

Iterative Augmentation (Sec 2.1)

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge$ $|x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x} \text{ is } \sqsubseteq\text{-minimal}\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *Graver-best step* for $\mathbf{x}$ if as good as $\lambda\mathbf{g}$ for any $\lambda \in \mathbb{N}$, $\mathbf{g} \in \mathcal{G}(A)$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *halfling* if half as good as Graver-best

Lm 7: Need $3n \log f_{\max}$ halflings for convergence (reduce $\approx \frac{1}{n}$ of gap by each step)

Lm 9: If $\mathbf{h}$ as good as any $\lambda\mathbf{g}$ for $\lambda \in \{1, 2, 4, 8, \ldots\}$ and $\mathbf{g} \in \mathcal{G}(A)$, then $\mathbf{h}$ is halfling

Iterative Augmentation (Sec 2.1)

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge$ $|x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ “closest to origin”)

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x}$ is $\sqsubseteq$ -minimal$\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *Graver-best step* for $\mathbf{x}$ if as good as $\lambda\mathbf{g}$ for any $\lambda \in \mathbb{N}$, $\mathbf{g} \in \mathcal{G}(A)$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *halfling* if half as good as Graver-best

Lm 7: Need $3n \log f_{\max}$ halflings for convergence (reduce $\approx \frac{1}{n}$ of gap by each step)

Lm 9: If $\mathbf{h}$ as good as any $\lambda\mathbf{g}$ for $\lambda \in \{1, 2, 4, 8, \dots\}$ and $\mathbf{g} \in \mathcal{G}(A)$, then $\mathbf{h}$ is halfling

Df: $\mathbf{g}$ solves $\mathcal{G}(A)$-best$\{f(\mathbf{x} + \lambda\mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda\mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathbb{Z}^n)\}$ if as good as

$$\min\{f(\mathbf{x} + \lambda\mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda\mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathcal{G}(A))\},$$

Iterative Augmentation (Sec 2.1)

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge |x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x}$ is $\sqsubseteq$ -minimal$\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *Graver-best step* for $\mathbf{x}$ if as good as $\lambda \mathbf{g}$ for any $\lambda \in \mathbb{N}$, $\mathbf{g} \in \mathcal{G}(A)$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *halfling* if half as good as Graver-best

Lm 7: Need $3n \log f_{\max}$ halflings for convergence (reduce $\approx \frac{1}{n}$ of gap by each step)

Lm 9: If $\mathbf{h}$ as good as any $\lambda \mathbf{g}$ for $\lambda \in \{1, 2, 4, 8, \ldots\}$ and $\mathbf{g} \in \mathcal{G}(A)$, then $\mathbf{h}$ is halfling

Df: $\mathbf{g}$ solves $\mathcal{G}(A)$-best$\{f(\mathbf{x} + \lambda \mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda \mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathbb{Z}^n)\}$ if as good as

$\min\{f(\mathbf{x} + \lambda \mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda \mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathcal{G}(A))\}$, call it (AugIP)

Iterative Augmentation (Sec 2.1)

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge |x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x} \text{ is } \sqsubseteq\text{-minimal}\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *Graver-best step* for $\mathbf{x}$ if as good as $\lambda\mathbf{g}$ for any $\lambda \in \mathbb{N}$, $\mathbf{g} \in \mathcal{G}(A)$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *halfling* if half as good as Graver-best

Lm 7: Need $3n \log f_{\max}$ halflings for convergence (reduce $\approx \frac{1}{n}$ of gap by each step)

Lm 9: If $\mathbf{h}$ as good as any $\lambda\mathbf{g}$ for $\lambda \in \{1, 2, 4, 8, \dots\}$ and $\mathbf{g} \in \mathcal{G}(A)$, then $\mathbf{h}$ is halfling

Df: $\mathbf{g}$ solves $\mathcal{G}(A)$-best$\{f(\mathbf{x} + \lambda\mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda\mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathbb{Z}^n)\}$ if as good as

$\min\{f(\mathbf{x} + \lambda\mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda\mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathcal{G}(A))\}$, call it (AugIP)

Lm 12: (AugIP) & $\mathbf{x}_0 \xRightarrow{\text{Lm 7 \& 9}}$ (IP) ($\approx n \log \|\mathbf{u} - \mathbf{l}\|_\infty \log f_{\max}$ calls to (AugIP) oracle)

($\mathbf{x} \sqsubseteq \mathbf{y} \Leftrightarrow \mathbf{x}$ and $\mathbf{y}$ in same orthant $\wedge$ $|x_i| \leq |y_i|$; $\mathbf{g} \in \mathcal{G}(A) \approx$ "closest to origin")

Definition (Graver basis)

$\mathcal{G}(A) = \{\mathbf{x} \in \mathrm{Ker}_{\mathbb{Z}}(A) \mid \mathbf{x}$ is $\sqsubseteq$ -minimal$\}$

Prop 6: Every $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ *conformally* decomposes as $\mathbf{h} = \sum_{i=1}^{2n} \lambda_i \mathbf{g}_i$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *Graver-best step* for $\mathbf{x}$ if as good as $\lambda\mathbf{g}$ for any $\lambda \in \mathbb{N}$, $\mathbf{g} \in \mathcal{G}(A)$

Df: $\mathbf{h} \in \mathrm{Ker}_{\mathbb{Z}}(A)$ is *halfling* if half as good as Graver-best

Lm 7: Need $3n \log f_{\max}$ halflings for convergence (reduce $\approx \frac{1}{n}$ of gap by each step)

Lm 9: If $\mathbf{h}$ as good as any $\lambda\mathbf{g}$ for $\lambda \in \{1, 2, 4, 8, \dots\}$ and $\mathbf{g} \in \mathcal{G}(A)$, then $\mathbf{h}$ is halfling

Df: $\mathbf{g}$ solves $\mathcal{G}(A)$-best$\{f(\mathbf{x} + \lambda\mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda\mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathbb{Z}^n)\}$ if as good as

$\min\{f(\mathbf{x} + \lambda\mathbf{g} \mid A\mathbf{g} = \mathbf{0},\ \mathbf{l} \leq \mathbf{x} + \lambda\mathbf{g} \leq \mathbf{u},\ \mathbf{g} \in \mathcal{G}(A))\}$, call it (AugIP)

Lm 12: (AugIP) & $\mathbf{x}_0 \xrightarrow{\text{Lm 7 \& 9}}$ (IP) ($\approx n \log \|\mathbf{u} - \mathbf{l}\|_\infty \log f_{\max}$ calls to (AugIP) oracle)

Lm 13: (AugIP) $\Longrightarrow$ $\mathbf{x}_0$ (auxiliary instance: get $\mathbf{x}$ s.t. $A\mathbf{x} = \mathbf{b}$, then minimize bound violation)

Df: $g_\infty(A) = \max_{\mathbf{g} \in \mathcal{G}(A)} \|\mathbf{g}\|_\infty, \;\; g_1(A) = \max_{\mathbf{g} \in \mathcal{G}(A)} \|\mathbf{g}\|_1$

Iterative Augmentation (Sections 2.3–2.4)

Df: $g_\infty(A) = \max_{\mathbf{g} \in \mathcal{G}(A)} \|\mathbf{g}\|_\infty$, $g_1(A) = \max_{\mathbf{g} \in \mathcal{G}(A)} \|\mathbf{g}\|_1$

Sec 2.3: Solving (AugIP) quickly by DP

Lm 22: (AugIP) solvable in time $(2g_\infty(A)+1)^{\mathrm{td}_P(A)} \cdot n$ (≈ CSP arc-consistency DP '85)

Lm 23: (AugIP) solvable in time $(2\|A\|_\infty g_1(A)+1)^{\mathcal{O}(\mathrm{td}_D(A))} \cdot n$ (≈ Papadimitriou's DP '81)

Iterative Augmentation (Sections 2.3–2.4)

Df: $g_\infty(A) = \max_{\mathbf{g}\in\mathcal{G}(A)} \|\mathbf{g}\|_\infty$, $g_1(A) = \max_{\mathbf{g}\in\mathcal{G}(A)} \|\mathbf{g}\|_1$

Sec 2.3: Solving (AugIP) quickly by DP

Lm 22: (AugIP) solvable in time $(2g_\infty(A)+1)^{\mathbf{td}_P(A)} \cdot n$ (≈ CSP arc-consistency DP ’85)

Lm 23: (AugIP) solvable in time $(2\|A\|_\infty g_1(A)+1)^{\mathcal{O}(\mathbf{td}_D(A))} \cdot n$ (≈ Papadimitriou’s DP ’81)

Sec 2.4: Bounding Graver norms

Lm 26: $g_\infty(A) \le g(\|A\|_\infty, \mathbf{td}_P(A))$ (g is exponential tower. Pf uses new lemma of Klein.)

Lm 28: $g_1(A) \le g(\|A\|_\infty, \mathbf{td}_D(A))$ (g is double-exp. Pf uses Steinitz lemma.)

Iterative Augmentation (Sections 2.3–2.4)

Df: $g_\infty(A) = \max_{\mathbf{g}\in\mathcal{G}(A)} \|\mathbf{g}\|_\infty$, $g_1(A) = \max_{\mathbf{g}\in\mathcal{G}(A)} \|\mathbf{g}\|_1$

Sec 2.3: Solving (AugIP) quickly by DP

Lm 22: (AugIP) solvable in time $(2g_\infty(A)+1)^{\mathrm{td}_P(A)} \cdot n$ (≈ CSP arc-consistency DP '85)

Lm 23: (AugIP) solvable in time $(2\|A\|_\infty g_1(A)+1)^{\mathcal{O}(\mathrm{td}_D(A))} \cdot n$ (≈ Papadimitriou's DP '81)

Sec 2.4: Bounding Graver norms (better for relevant special cases)

Lm 26: $g_\infty(A) \le g(\|A\|_\infty, \mathrm{td}_P(A))$ (g is exponential tower. Pf uses new lemma of Klein.)

Lm 28: $g_1(A) \le g(\|A\|_\infty, \mathrm{td}_D(A))$ (g is double-exp. Pf uses Steinitz lemma.)

Iterative Augmentation (Sections 2.3–2.4)

Df: $g_\infty(A) = \max_{\mathbf{g}\in\mathcal{G}(A)} \|\mathbf{g}\|_\infty$, $g_1(A) = \max_{\mathbf{g}\in\mathcal{G}(A)} \|\mathbf{g}\|_1$

Sec 2.3: Solving (AugIP) quickly by DP

Lm 22: (AugIP) solvable in time $(2g_\infty(A)+1)^{\mathrm{td}_P(A)} \cdot n$ (≈ CSP arc-consistency DP ’85)

Lm 23: (AugIP) solvable in time $(2\|A\|_\infty g_1(A)+1)^{\mathcal{O}(\mathrm{td}_D(A))} \cdot n$ (≈ Papadimitriou’s DP ’81)

Sec 2.4: Bounding Graver norms (better for relevant special cases)

Lm 26: $g_\infty(A) \le g(\|A\|_\infty, \mathrm{td}_P(A))$ (g is exponential tower. Pf uses new lemma of Klein.)

Lm 28: $g_1(A) \le g(\|A\|_\infty, \mathrm{td}_D(A))$ (g is double-exp. Pf uses Steinitz lemma.)

Theorem

(IP) solvable in time $g(\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}, \|A\|_\infty) n^2 \log \|\mathbf{u}-\mathbf{l}\|_\infty \log f_{\max}$.

The Extras

The Extras: Proximity Bounds

Theorem (Basic proximity)

Let $\mathbf{x}^*, \mathbf{z}^*$ *be a fractional and integer optimum, respectively. There exist* $\hat{\mathbf{x}}, \hat{\mathbf{z}}$ *frac/int optima s.t., for any* $p \geq 1$,

$$\|\mathbf{x}^* - \hat{\mathbf{z}}\|_p, \|\mathbf{z}^* - \hat{\mathbf{x}}\|_p \leq n \cdot g_p(A) \quad .$$

The Extras: Proximity Bounds

Theorem (Basic proximity)

Let $\mathbf{x}^*, \mathbf{z}^*$ *be a fractional and integer optimum, respectively. There exist* $\hat{\mathbf{x}}, \hat{\mathbf{z}}$ *frac/int optima s.t., for any* $p \geq 1$,

$$\|\mathbf{x}^* - \hat{\mathbf{z}}\|_p, \|\mathbf{z}^* - \hat{\mathbf{x}}\|_p \leq n \cdot g_p(A) \quad .$$

Theorem (Scaling proximity)

Let $\mathbf{x}^s$ *be opt of* s*-scaled down instance. There exists* $\mathbf{x}^*$ *opt s.t.*

$$\|s \cdot \mathbf{x}^s - \mathbf{x}^*\|_p \leq s \cdot n \cdot g_p(A) \quad .$$

The Extras: Proximity Bounds

Theorem (Basic proximity)

Let $\mathbf{x}^*, \mathbf{z}^*$ be a fractional and integer optimum, respectively. There exist $\hat{\mathbf{x}}, \hat{\mathbf{z}}$ frac/int optima s.t., for any $p \geq 1$,

$$\|\mathbf{x}^* - \hat{\mathbf{z}}\|_p, \|\mathbf{z}^* - \hat{\mathbf{x}}\|_p \leq n \cdot g_p(A) \quad .$$

Theorem (Scaling proximity)

Let $\mathbf{x}^s$ be opt of s-scaled down instance. There exists $\mathbf{x}^*$ opt s.t.

$$\|s \cdot \mathbf{x}^s - \mathbf{x}^*\|_p \leq s \cdot n \cdot g_p(A) \quad .$$

“Modern version” of [Hochbaum, Shantikumar ’90]

The Extras: Reducibility Bounds

Runtime dependence on $\log f_{\max} \approx$ obstacle for strongly-poly algos

The Extras: Reducibility Bounds

Runtime dependence on $\log f_{\max} \approx$ obstacle for strongly-poly algos

$\implies$ Replace $\mathbf{w}\mathbf{x}$ with $\mathbf{w}'\mathbf{x}$ which is *equivalent* (does not change optima) and $\|\mathbf{w}'\|_\infty \leq 2^{\mathrm{poly}(N,n)}$ if $\|\mathbf{u}, \boldsymbol{\ell}\|_\infty \leq N$. [Frank, Tardos '87]

The Extras: Reducibility Bounds

Runtime dependence on $\log f_{\max} \approx$ obstacle for strongly-poly algos

$\implies$ Replace $\mathbf{w}\mathbf{x}$ with $\mathbf{w}'\mathbf{x}$ which is *equivalent* (does not change optima) and $\|\mathbf{w}'\|_\infty \leq 2^{\mathrm{poly}(N,n)}$ if $\|\mathbf{u}, \boldsymbol{\ell}\|_\infty \leq N$. [Frank, Tardos '87]

BUT: adds $n^3 \log n$ factor in the runtime :(

The Extras: Reducibility Bounds

Runtime dependence on $\log f_{\max}$ ≈ obstacle for strongly-poly algos

$\implies$ Replace $\mathbf{w}\mathbf{x}$ with $\mathbf{w}'\mathbf{x}$ which is *equivalent* (does not change optima) and $\|\mathbf{w}'\|_\infty \leq 2^{\mathrm{poly}(N,n)}$ if $\|\mathbf{u},\boldsymbol{\ell}\|_\infty \leq N$. [Frank, Tardos '87]

BUT: adds $n^3 \log n$ factor in the runtime :(

Theorem (Linear reducibility)

1) $\exists$ equivalent $\mathbf{w}'$ s.t. $\|\mathbf{w}'\|_\infty \leq N^n$, *2) asymptotically optimal*

Runtime dependence on $\log f_{\max} \approx$ obstacle for strongly-poly algos

$\implies$ Replace $\mathbf{w}\mathbf{x}$ with $\mathbf{w}'\mathbf{x}$ which is *equivalent* (does not change optima) and $\|\mathbf{w}'\|_\infty \leq 2^{\text{poly}(N,n)}$ if $\|\mathbf{u}, \boldsymbol{\ell}\|_\infty \leq N$. [Frank, Tardos '87]

BUT: adds $n^3 \log n$ factor in the runtime :(

Theorem (Linear reducibility)

1) $\exists$ *equivalent* $\mathbf{w}'$ *s.t.* $\|\mathbf{w}'\|_\infty \leq N^n$, *2) asymptotically optimal*

Theorem (Separable-convex reducibility)

1) $\exists$ *equivalent* f' *s.t.* $f'_{\max} \leq (n^2 N)^{nN}$, *2) asymptotically optimal*

The Extras: Strongly-poly Algorithm

Goal: Strongly-polynomial algorithm (#arithmetic ops not dep on size of numbers)

1. Solve LP relaxation in $\mathrm{poly}(n \cdot \langle A \rangle)$ time [Tardos '86]
2. Proximity: int opt not far from frac opt $\Rightarrow$ shrink bounds $\mathbf{l}', \mathbf{u}'$, shrink rhs $\mathbf{b}'$.
3. Reduce objective: $\mathbf{l}', \mathbf{u}'$ give small box $\Rightarrow$ equiv. $\mathbf{w}'$ w/ small $\|\mathbf{w}'\|_\infty$
4. Convergence: $3n\langle A, \mathbf{w}', \mathbf{b}', \mathbf{l}', \mathbf{u}' \rangle = \mathrm{poly}(n \cdot \langle A \rangle)$ halflings reach optimum.

The Extras: Strongly-poly Algorithm

Goal: Strongly-polynomial algorithm (#arithmetic ops not dep on size of numbers)

1. Solve LP relaxation in poly($n \cdot \langle A \rangle$) time [Tardos '86]
2. Proximity: int opt not far from frac opt $\Rightarrow$ shrink bounds $\mathbf{l}', \mathbf{u}'$, shrink rhs $\mathbf{b}'$.
3. Reduce objective: $\mathbf{l}', \mathbf{u}'$ give small box $\Rightarrow$ equiv. $\mathbf{w}'$ w/ small $\|\mathbf{w}'\|_\infty$
4. Convergence: $3n\langle A, \mathbf{w}', \mathbf{b}', \mathbf{l}', \mathbf{u}' \rangle = \text{poly}(n \cdot \langle A \rangle)$ halflings reach optimum.

The Extras: Strongly-poly Algorithm

Goal: Strongly-polynomial algorithm (#arithmetic ops not dep on size of numbers)

1. Solve LP relaxation in poly($n \cdot \langle A \rangle$) time [Tardos '86]
2. Proximity: int opt not far from frac opt $\Rightarrow$ shrink bounds $\mathbf{l}', \mathbf{u}'$, shrink rhs $\mathbf{b}'$.
3. Reduce objective: $\mathbf{l}', \mathbf{u}'$ give small box $\Rightarrow$ equiv. $\mathbf{w}'$ w/ small $\|\mathbf{w}'\|_\infty$
4. Convergence: $3n\langle A, \mathbf{w}', \mathbf{b}', \mathbf{l}', \mathbf{u}' \rangle = \text{poly}(n \cdot \langle A \rangle)$ halflings reach optimum.

The Extras: Strongly-poly Algorithm

Goal: Strongly-polynomial algorithm (#arithmetic ops not dep on size of numbers)

1. Solve LP relaxation in poly($n \cdot \langle A \rangle$) time [Tardos '86]
2. Proximity: int opt not far from frac opt $\Rightarrow$ shrink bounds $\mathbf{l}', \mathbf{u}'$, shrink rhs $\mathbf{b}'$.
3. Reduce objective: $\mathbf{l}', \mathbf{u}'$ give small box $\Rightarrow$ equiv. $\mathbf{w}'$ w/ small $\|\mathbf{w}'\|_\infty$
4. Convergence: $3n\langle A, \mathbf{w}', \mathbf{b}', \mathbf{l}', \mathbf{u}' \rangle = \text{poly}(n \cdot \langle A \rangle)$ halflings reach optimum.

The Extras: Strongly-poly Algorithm

Goal: Strongly-polynomial algorithm (#arithmetic ops not dep on size of numbers)

1. Solve LP relaxation in $\mathrm{poly}(n \cdot \langle A \rangle)$ time [Tardos '86]
2. Proximity: int opt not far from frac opt $\Rightarrow$ shrink bounds $\mathbf{l}', \mathbf{u}'$, shrink rhs $\mathbf{b}'$.
3. Reduce objective: $\mathbf{l}', \mathbf{u}'$ give small box $\Rightarrow$ equiv. $\mathbf{w}'$ w/ small $\|\mathbf{w}'\|_\infty$
4. Convergence: $3n\langle A, \mathbf{w}', \mathbf{b}', \mathbf{l}', \mathbf{u}' \rangle = \mathrm{poly}(n \cdot \langle A \rangle)$ halflings reach optimum.

Theorem

ILP solvable in time $g(\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}, \|A\|_\infty)\, poly(n)$.

The Extras: Near-linear Algorithms

Goal: Improve n^2 to $n \operatorname{poly} \log n$.

The Extras: Near-linear Algorithms

Goal: Improve n^2 to $n \operatorname{poly} \log n$.

Small $\mathrm{td}_P(A)$

- Replace halflings with a more expensive, more powerful steps
  $\implies g_\infty(A) \cdot \log f_{\max}$ steps convergence (instead of $n \cdot \log f_{\max}$)
- Need all tricks to smooth remaining issues (proximity-scaling, reducibility, ...)

The Extras: Near-linear Algorithms

Goal: Improve n^2 to $n\,\text{poly}\log n$.

Small $\text{td}_P(A)$

- Replace halflings with a more expensive, more powerful steps
  $\implies$ $g_\infty(A) \cdot \log f_{\max}$ steps convergence (instead of $n \cdot \log f_{\max}$)
- Need all tricks to smooth remaining issues (proximity-scaling, reducibility, ...)

Small $\text{td}_D(A)$

- 👁: since $g_1(A)$ small, only few coordinates of $\mathbf{x}$ changed in each step
- $\implies$ construct a data structure computing (AugIP) which updates in $\log n$ time after a halfling step

The Extras: Near-linear Algorithms

Goal: Improve n^2 to n poly $\log n$.

Small $\mathrm{td}_P(A)$

- Replace halflings with a more expensive, more powerful steps
  $\implies g_\infty(A) \cdot \log f_{\max}$ steps convergence (instead of $n \cdot \log f_{\max}$)
- Need all tricks to smooth remaining issues (proximity-scaling, reducibility, ...)

Small $\mathrm{td}_D(A)$

- 👁: since $g_1(A)$ small, only few coordinates of $\mathbf{x}$ changed in each step
- $\implies$ construct a data structure computing (AugIP) which updates in $\log n$ time after a halfling step

Theorem ((more or less))

(IP) solvable in time $g(\min\{\mathrm{td}_P(A), \mathrm{td}_D(A)\}, \|A\|_\infty) n \log^{\mathcal{O}(1)} n \log \|\mathbf{u} - \mathbf{l}\|_\infty \log f_{\max}$.

Outlook

Outlook

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Don't be afraid of the paper!

Outlook

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Don’t be afraid of the paper!

- Other results:
  - ETH-based lower bounds
  - Applications: scheduling, bin packing, computational social choice, …

Outlook

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Don't be afraid of the paper!

- Other results:
  - ETH-based lower bounds
  - Applications: scheduling, bin packing, computational social choice, ...
- New directions:
  - Mixed-integer? Yes when $f(\mathbf{x}) = \mathbf{w}\mathbf{x}$, sep-convex open.

Outlook

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Don't be afraid of the paper!

- Other results:
  - ETH-based lower bounds
  - Applications: scheduling, bin packing, computational social choice, ...
- New directions:
  - Mixed-integer? Yes when $f(\mathbf{x}) = \mathbf{wx}$, sep-convex open.
  - Implementation: DP is the main obstacle. **We need help!**

Outlook

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Don't be afraid of the paper!

- Other results:
  - ETH-based lower bounds
  - Applications: scheduling, bin packing, computational social choice, …
- New directions:
  - Mixed-integer? Yes when $f(\mathbf{x}) = \mathbf{w}\mathbf{x}$, sep-convex open.
  - Implementation: DP is the main obstacle. **We need help!**
  - Row-invariant parameters: branch-depth, …?

Outlook

An Algorithmic Theory of Integer Programming [arxiv:1904.01361]

Don't be afraid of the paper!

- Other results:
  - ETH-based lower bounds
  - Applications: scheduling, bin packing, computational social choice, …
- New directions:
  - Mixed-integer? Yes when $f(\mathbf{x}) = \mathbf{w}\mathbf{x}$, sep-convex open.
  - Implementation: DP is the main obstacle. **We need help!**
  - Row-invariant parameters: branch-depth, …?

Thank you!