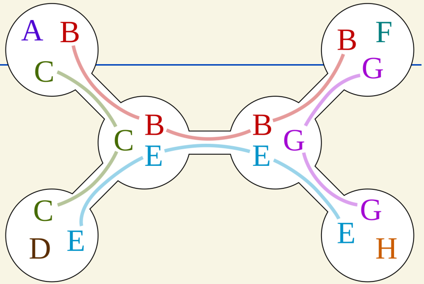
FPT via Linear Programming II



DM898: Parameterized Algorithms
Lars Rohwedder

Today's lecture

FPT algorithms for classes of integer linear programs

- Parameter: number of variables
- Parameter: number of constraints

FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

parameters need to be strong enough to counter high difficulty of ILPs

FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

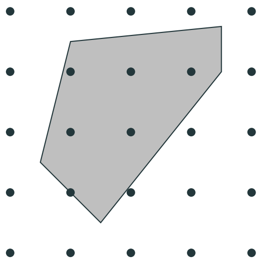
parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

Idea based on branching:

- We want to decide if there is a point that is in the feasible region of the LP relaxation (a polytope) and is integer



FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

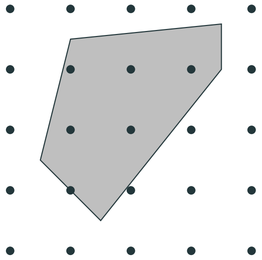
parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

Idea based on branching:

- We want to decide if there is a point that is in the feasible region of the LP relaxation (a polytope) and is integer
- If the polytope is not "flat" in any direction, it must contain an integer point $\rightsquigarrow$ return YES



FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

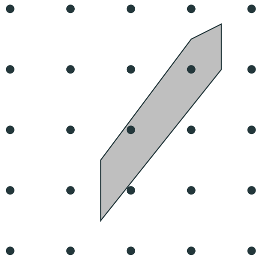
parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

Idea based on branching:

- We want to decide if there is a point that is in the feasible region of the LP relaxation (a polytope) and is integer
- If the polytope is not "flat" in any direction, it must contain an integer point $\rightsquigarrow$ return YES
- Otherwise, find flat direction and branch on a small number of orthogonal hyperplanes on which an integer point can lie $\rightsquigarrow$ new problem is in dimension $n-1$



FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

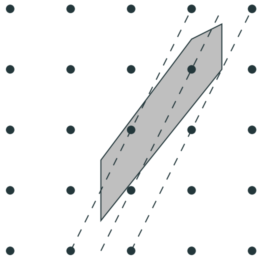
parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

Idea based on branching:

- We want to decide if there is a point that is in the feasible region of the LP relaxation (a polytope) and is integer
- If the polytope is not "flat" in any direction, it must contain an integer point $\rightsquigarrow$ return YES
- Otherwise, find flat direction and branch on a small number of orthogonal hyperplanes on which an integer point can lie $\rightsquigarrow$ new problem is in dimension $n-1$



Details are intricate and we do not cover them here

FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

Eisenbrand-Weismantel algorithm

Eisenbrand and Weismantel [2018] showed that integer programs with m linear equality constraints, where all coefficients lie in $\{-\Delta, \ldots, \Delta\}$, and (possibly many) variables with upper and lower bounds can be solved in time $(m\Delta)^{O(m^2)} \cdot |I|^{O(1)}$.

FPT algorithms for classes of ILPs

Can we design FPT algorithms for integer linear programs?

Motivation: since ILPs are very expressive, we would be able to derive many results from such an algorithm

But how to parameterize ILPs?

parameters need to be strong enough to counter high difficulty of ILPs

Lenstra's algorithm

Lenstra [1983] showed that ILPs with n variables can be solved in time $f(n) \cdot |I|^{O(1)}$.
Subsequent improvements by Kannan [1987] and Reis and Rothvoss [2023] improved the running time to $(\log n)^{O(n)} \cdot |I|^{O(1)}$.

Eisenbrand-Weismantel algorithm

Eisenbrand and Weismantel [2018] showed that integer programs with m linear equality constraints, where all coefficients lie in $\{-\Delta, \dots, \Delta\}$, and (possibly many) variables with upper and lower bounds can be solved in time $(m\Delta)^{O(m^2)} \cdot |I|^{O(1)}$.

One may ask whether Eisenbrand and Weismantel's algorithm can be improved to FPT time in m alone (like Lentra's algorithm that does not need parameter Δ). This is not possible, because $m = 1$ is already NP-hard (e.g. via reduction from Subset-Sum).

We will prove Eisenbrand and Weismantel's result in the following.

Eisenbrand-Weismantel algorithm

Proximity

Problem statement. Given coefficients of the objective $c \in \mathbb{Z}^n$, a matrix $A \in \{-\Delta, \ldots, \Delta\}^{m \times n}$ (encoding the coefficients of the constraints in the m rows), right-hand side $b \in \mathbb{Z}^m$ and lower and upper bounds $\ell_i \in \mathbb{Z} \cup \{-\infty\}, u_i \in \mathbb{Z} \cup \{\infty\}$, $i \in \{1, 2, \ldots, n\}$, solve

$$\begin{aligned} &\min\ c^\mathsf{T} x \\ &Ax = b \\ &\ell_i \le x_i \le u_i, \quad x_i \in \mathbb{Z} && \text{for all } i = 1, 2, \ldots, n \end{aligned}$$

Proximity

Problem statement. Given coefficients of the objective $c \in \mathbb{Z}^n$, a matrix $A \in \{-\Delta, \ldots, \Delta\}^{m \times n}$ (encoding the coefficients of the constraints in the m rows), right-hand side $b \in \mathbb{Z}^m$ and lower and upper bounds $\ell_i \in \mathbb{Z} \cup \{-\infty\}, u_i \in \mathbb{Z} \cup \{\infty\}$, $i \in \{1, 2, \ldots, n\}$, solve

$$\begin{aligned} &\min\ c^\mathsf{T} x \\ &Ax = b \\ &\ell_i \le x_i \le u_i, \quad x_i \in \mathbb{Z} && \text{for all } i = 1, 2, \ldots, n \end{aligned}$$

Proximity theorem

Assume that the ILP above is feasible and bounded. Let x^* be a optimal solution to the LP relaxation of the ILP above with at most m non-integral variables[1]. Then there exists some optimal integer solution x with

$$\|x - x^*\|_1 = \sum_{i=1}^{n} |x_i - x_i^*| \le (2m^2\Delta + 1)^m + m =: \text{prox}$$

$$\left[\quad \text{in particular:} \quad \|x - \lfloor x^* \rfloor\|_1 = \sum_{i=1}^{n} |x_i - \lfloor x_i^* \rfloor| \le (2m^2\Delta + 1)^m + 2m =: \text{prox}' \quad \right]$$

We will prove this statement next lecture and first show how it can be used algorithmically.

---

[1] such a solution always exists and can be computed in polynomial time. More details next lecture.

Dynamic program

We proceed similar to the Knapsack dynamic program based on "dominance"

ILP(n, A, b, c, ℓ, u)

- compute optimum x^* to LP relaxation with $\leq m$ non-integral variables
- $\mathcal{T} \leftarrow \{(0,0,0)\}$ // set of undominated (objective, right-hand side, distance-to-$\lfloor x^* \rfloor$) triples obtainable
- for $i \in \{1, 2, \ldots, n\}$
  - $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
  - for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \ldots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
    - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x_i^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
  - for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C'$, $B = B'$, $k = k'$
    - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
  - for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
    - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$
- return $\min\{C \mid (C, B, k) \in \mathcal{T}, B = b\}$

Dynamic program

We proceed similar to the Knapsack dynamic program based on "dominance"

ILP(n, A, b, c, ℓ, u)

- compute optimum x^* to LP relaxation with $\leq m$ non-integral variables
- $\mathcal{T} \leftarrow \{(0,0,0)\}$ // set of undominated (objective, right-hand side, distance-to-$\lfloor x^* \rfloor$) triples obtainable
- for $i \in \{1, 2, \dots, n\}$
  - $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
  - for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox}'\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox}'\}\}$
    - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x_i^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
  - for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C'$, $B = B'$, $k = k'$
    - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
  - for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
    - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$
- return $\min\{C \mid (C, B, k) \in \mathcal{T}, B = b\}$

Correctness. By induction we have that for each $B \in \mathbb{Z}^m$, $k \in \mathbb{Z}_{\geq 0}$, $\mathcal{T}$ contains (C, B, k) after iteration i if and only if $k \leq$ prox' and the following minimum exists with

$$C = \min\left\{\sum_{j=1}^{i} c_j x_j \mid x_i \in \{\ell_i, \ell_i + 1, \dots, u_i\} \forall j \in \{1, 2 \dots, i\}, \sum_{j=1}^{i} A_j x_j = B, \sum_{j=1}^{i} |x_j - \lfloor x_j^* \rfloor| = k\right\}$$

Running time

One iteration of main loop

- $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
- for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
  - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
- for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C', B = B'$, $k = k'$
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
- for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$

The iteration has running time $(\text{prox'} \cdot |\mathcal{T}|)^{O(1)}$ with $|\mathcal{T}|$ at the beginning of iteration. **How large is $\mathcal{T}$?**

Running time

One iteration of main loop

- $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
- for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
  - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
- for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C', B = B'$, $k = k'$
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
- for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$

The iteration has running time $(\text{prox'} \cdot |\mathcal{T}|)^{O(1)}$ with $|\mathcal{T}|$ at the beginning of iteration. **How large is $\mathcal{T}$?**

- At most one tripel (C, B, k) for each B, k

Running time

One iteration of main loop

- $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
- for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
  - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
- for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C', B = B'$, $k = k'$
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
- for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$

The iteration has running time $(\text{prox'} \cdot |\mathcal{T}|)^{O(1)}$ with $|\mathcal{T}|$ at the beginning of iteration. **How large is $\mathcal{T}$?**

- At most one tripel (C, B, k) for each B, k
- Many B, k have **no** tripel. If there is a tripel (C, B, k), then $k \leq$ prox' and there is a solution $x_1, \dots, x_i$ with $A_1 x_1 + \cdots A_i x_i = B$ and $\sum_{j=1}^{i} |x_j - \lfloor x_j^* \rfloor| = k$. Thus,

$$\|B - \underbrace{(A_1 \lfloor x_1^* \rfloor + \cdots + A_i \lfloor x_i^* \rfloor)}_{:= B^*}\|_\infty \leq k\Delta \ .$$

Running time

One iteration of main loop

- $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
- for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
  - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
- for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C', B = B'$, $k = k'$
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
- for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$

The iteration has running time $(\text{prox'} \cdot |\mathcal{T}|)^{O(1)}$ with $|\mathcal{T}|$ at the beginning of iteration. **How large is $\mathcal{T}$?**

- At most one tripel (C, B, k) for each B, k
- Many B, k have **no** tripel. If there is a tripel (C, B, k), then $k \leq$ prox' and there is a solution $x_1, \dots, x_i$ with $A_1 x_1 + \cdots A_i x_i = B$ and $\sum_{j=1}^{i} |x_j - \lfloor x_j^* \rfloor| = k$. Thus,

$$\|B - \underbrace{(A_1 \lfloor x_1^* \rfloor + \cdots + A_i \lfloor x_i^* \rfloor)}_{:= B^*}\|_\infty \leq k\Delta \ .$$

  But there only exist $(2k\Delta + 1)^m \leq (2\text{prox'}\Delta + 1)^m$ integer vectors B with $\|B - B^*\|_\infty \leq k\Delta$.

Running time

One iteration of main loop

- $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
- for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
  - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
- for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C', B = B'$, $k = k'$
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
- for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$

The iteration has running time $(\text{prox'} \cdot |\mathcal{T}|)^{O(1)}$ with $|\mathcal{T}|$ at the beginning of iteration. **How large is $\mathcal{T}$?**

- At most one tripel (C, B, k) for each B, k
- Many B, k have **no** tripel. If there is a tripel (C, B, k), then $k \leq$ prox' and there is a solution $x_1, \dots, x_i$ with $A_1 x_1 + \cdots A_i x_i = B$ and $\sum_{j=1}^{i} |x_j - \lfloor x_j^* \rfloor| = k$. Thus,
$$\|B - \underbrace{(A_1 \lfloor x_1^* \rfloor + \cdots + A_i \lfloor x_i^* \rfloor)}_{:= B^*}\|_\infty \leq k\Delta \ .$$
  But there only exist $(2k\Delta + 1)^m \leq (2\text{prox'}\Delta + 1)^m$ integer vectors B with $\|B - B^*\|_\infty \leq k\Delta$.
- Thus, at the beginning of an iteration $|\mathcal{T}| \leq (2\text{prox'}\Delta + 1)^m \cdot \text{prox'} \leq (m\Delta)^{O(m^2)}$

Running time

One iteration of main loop

- $\mathcal{T}' \leftarrow \mathcal{T}$, $\mathcal{T} \leftarrow \emptyset$
- for x_i in $\{\max\{\ell_i, \lfloor x_i^* \rfloor - \text{prox'}\}, \dots, \min\{u_i, \lfloor x_i^* \rfloor + \text{prox'}\}\}$
  - $\mathcal{T} \leftarrow \mathcal{T} \cup \{(C + c_i x_i, B + A_i x_i, k + |\lfloor x^* \rfloor - x_i|) \mid (C, B, k) \in \mathcal{T}'\}$ // A_i is the ith column of A
- for $(C, B, k), (C', B', k') \in \mathcal{T}$ with $C < C', B = B'$, $k = k'$
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C', B', k')$
- for $(C, B, k) \in \mathcal{T}$ with $k >$ prox'
  - $\mathcal{T} \leftarrow \mathcal{T} \setminus (C, B, k)$

The iteration has running time $(\text{prox'} \cdot |\mathcal{T}|)^{O(1)}$ with $|\mathcal{T}|$ at the beginning of iteration. **How large is $\mathcal{T}$?**

- At most one tripel (C, B, k) for each B, k
- Many B, k have **no** tripel. If there is a tripel (C, B, k), then $k \leq$ prox' and there is a solution $x_1, \dots, x_i$ with $A_1 x_1 + \cdots A_i x_i = B$ and $\sum_{j=1}^{i} |x_j - \lfloor x_j^* \rfloor| = k$. Thus,

$$\|B - \underbrace{(A_1 \lfloor x_1^* \rfloor + \cdots + A_i \lfloor x_i^* \rfloor)}_{:= B^*}\|_\infty \leq k\Delta \ .$$

  But there only exist $(2k\Delta + 1)^m \leq (2\text{prox'}\Delta + 1)^m$ integer vectors B with $\|B - B^*\|_\infty \leq k\Delta$.
- Thus, at the beginning of an iteration $|\mathcal{T}| \leq (2\text{prox'}\Delta + 1)^m \cdot \text{prox'} \leq (m\Delta)^{O(m^2)}$

Total running time: #iterations $\cdot\, (m\Delta)^{O(m^2)} \leq n \cdot (m\Delta)^{O(m^2)}$

Other FPT results for ILP

Unbounded variables

Consider the Eisenbrand-Weismantel setting, but with all variables being non-negative integers:

$$\begin{aligned} &\min\ c^\intercal x \\ &Ax = b \\ &x_i \in \mathbb{Z}_{\geq 0} \qquad\qquad \text{for all } i = 1, 2, \ldots, n \end{aligned}$$

The Eisenbrand-Weismantel algorithm from before has a running time of $n(m\Delta)^{O(m^2)}$. In this case, faster algorithms are known.

Fastest known algorithm runs in time $(\sqrt{m}\Delta)^{2m} + O(nm)$, due to Jansen and Rohwedder [2019].

Block structures

Consider $A_1, \dots, A_n, B_1, \dots, B_n \in \{-\Delta, \dots, \Delta\}^{k \times k}$ that form one of the following two block structures.

$$
\begin{aligned}
&\min c^\mathsf{T} x \\
&\begin{pmatrix}
A_1 & A_2 & \cdots & A_n \\
B_1 & & & \\
& B_2 & & \\
& & \ddots & \\
& & & B_n
\end{pmatrix} x = b \\
&\ell_i \le x_i \le u_i \quad \text{for all } i = 1, 2, \dots, kn \\
&x_i \in \mathbb{Z}
\end{aligned}
$$

$$
\begin{aligned}
&\min c^\mathsf{T} x \\
&\begin{pmatrix}
A_1 & B_1 & & & \\
A_2 & & B_2 & & \\
\vdots & & & \ddots & \\
A_n & & & & B_n
\end{pmatrix} x = b \\
&\ell_i \le x_i \le u_i \quad \text{for all } i = 1, 2, \dots, k(n+1) \\
&x_i \in \mathbb{Z}
\end{aligned}
$$

Both classes of ILPs have FPT algorithms in parameters k and Δ. Currently fastest are due to Cslovjecsek, Eisenbrand, Hunkenschröder, Rohwedder, Weismantel [2021] and Klein [2020].

In **practice**, these classes are also well solvable via decomposition methods (Dantzig-Wolfe decomposition and Bender's decomposition), which we do not detail here.

Summary

A number of FPT results for integer linear programs are known, which:

- Explain to some extend the good empirical behavior and utility of LP relaxations we can see in commercial ILP solvers
- Have some interesting applications for concrete problems (see e.g. exercises)
- **Disclaimer:** Many (perhaps most) FPT results in literature are based on entirely different techniques and cannot be derived from generic ILP results

Summary

A number of FPT results for integer linear programs are known, which:

- Explain to some extend the good empirical behavior and utility of LP relaxations we can see in commercial ILP solvers
- Have some interesting applications for concrete problems (see e.g. exercises)
- **Disclaimer:** Many (perhaps most) FPT results in literature are based on entirely different techniques and cannot be derived from generic ILP results

PARAMLP

- ERC-funded project PARAMLP: Parameterized Algorithms and Polyhedra
- Funded with $\approx 11\,000\,000$ DKK
- Awarded to me in 2025, hiring PhD students and PostDocs soon
- Possibly also thesis projects in this field