École Polytechnique Fédérale de Lausanne
Lausanne | Vaud | Switzerland

Institute of Mathematics

College of Basic Sciences

Week 4: Notes

— MATH-504 | Integer Optimisation | (Spring 2023)

By:
Edmund Hofflin

Email:
edmund.hofflin@epfl.ch
Contact me if there are any mistakes or issues.

Professor:
Prof. Friedrich Eisenbrand

March 2023

3 Dynamic Programming I

Definition 3.1 (Knapsack Problem)**.** The *Knapsack Problem* is as follows: Given $n \in \mathbb{N}$ items, each with weight $a_i \in Z_+$ and profit $p_i \in \mathbb{Z}_+$, and a maximum capacity $D \in \mathbb{Z}_+$, we wish to find $S \subseteq [n]$ such that $\sum_{i \in S} a_i \leq D$ and $\sum_{i \in S} p_i$ is maximal. That is, we wish to find the collection of items that are below the maximum capacity and yield the most profit.

Proposition 3.1 (Knapsack as IP)**.** *All knapsack problems can be represented as IPs.*

Proof. By construction:

$$\max \mathbf{p}^\top \mathbf{x}$$
$$s.t\ \mathbf{a}^\top \mathbf{x} \leq D, \mathbf{x} \in \{0,1\}^n$$

where $\mathbf{p}, \mathbf{a} \in \mathbb{N}^n$ and $D \in \mathbb{N}$. $\mathbf{x}_i$ is active if item i is chosen. □

Example 3.1 (Simple Knapsack)**.** The following is a Knapsack problem:

$$\max 10\mathbf{x}_1 + 5\mathbf{x}_2 + 8\mathbf{x}_3$$
$$s.t\ 3\mathbf{x}_1 + 2\mathbf{x}_2 + \mathbf{x}_3 \leq 5, \mathbf{x} \in \{0,1\}^3$$

Proposition 3.2 (Acyclic Graphs and Knapsacks)**.** *For a given knapsack problem, an assignment of $\mathbf{x} \in \mathbb{R}^n$ corresponds to a path in the acyclic graph*

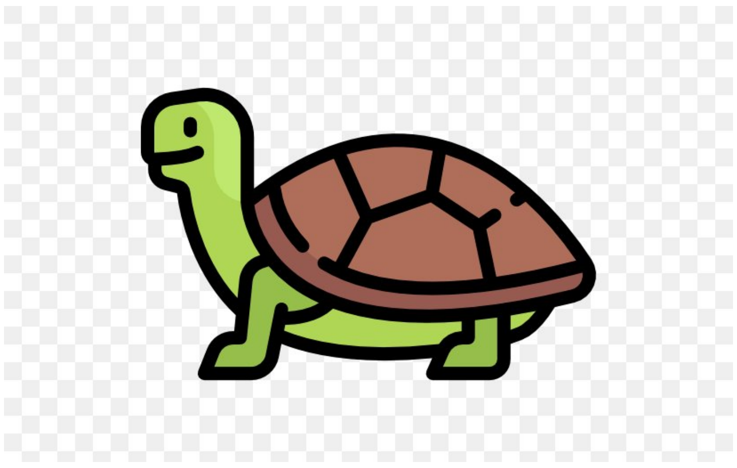

Figure 1: Acyclic Graph for Knapsack Problem

where $(0,0)$ is the initial node and all final nodes are (n, w) for some $w \leq D$. A path from the initial node to a final node corresponds to a selection of items with summed weight not larger than D. Note that $|V| \leq nD$ in this graph.

Proposition 3.3 (Optimal Solutions are Longest Paths)**.** *For a given knapsack problem, an optimal solution $\mathbf{x}^* \in \mathbb{R}^n$ corresponds to a longest path in the acyclic graph.*

Example 3.2 (Simple Knapsack Graph)**.** For the knapsack problem in Example 3.1, the corresponding acyclic graph is:

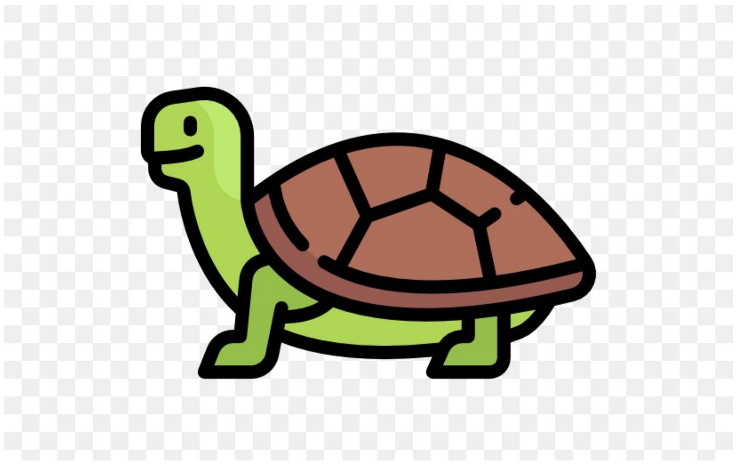

Figure 2: Acyclic Graph for Simple Knapsack

The longest path is clearly $\mathbf{x} = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}^\top$, with total weight 4 and profit 18.

Proposition 3.4 (Solving Knapsack through Generating Table). *Let $A : [n] \times [D] \to \mathbb{N}$ be defined by*

$$A(i, w) = \begin{cases} \max\left\{A(i-1, w), A(i-1, w-a_i) + p_i\right\}, & w - a_i \geq 0 \\ A(i-1, w), & \textit{otherwise} \end{cases}$$

where $A(0,0) = 0$. Generating $A(n, D)$ is $\mathcal{O}(nD)$.

Proof. Generating $A(n, D)$ corresponds to moving backwards through the acyclic graph to find the longest path. □

Proposition 3.5 (Solving Knapsack through Generating Table Alternative). *Let $B : [n] \times \mathbb{N} \to [D]$ be defined through*

$$B(i, p) = \begin{cases} \min\left\{B(i-1, p), B(i-1, p-p_i) + w_i\right\}, & p - p_i > 0 \\ B(i-1, p), & \textit{otherwise} \end{cases}$$

where $B(0,0) = 0$. Generating $B(n, n\mathbf{p}_{\max})$ is $\mathcal{O}(n^2 \|\mathbf{p}\|_\infty)$, where $\mathbf{p}_{\max} = \|\mathbf{p}\|_\infty$.

Proof. Generating $B(n, np_{\max})$ corresponds to finding the minimum weight for a subset of $[n]$ to achieve profit p. □

Definition 3.2 (Pseudo-Polynomial)**.** A algorithm has *Pseudo-Polynomial* complexity with respect to un-encoded (i.e. not in binary) input n iff it is $\mathcal{O}(p(n))$ for some polynomial p. Note that when encoded, a pseudo-polynomial algorithm has exponential complexity with respect to the length of the input.

Example 3.3 (Knapsack Complexity). Propositions 3.4 and 3.5 show that Knapsack has pseudo-polynomial complexity with respect to D and $\mathbf{p}_{\max}$.

Theorem 3.1 (Approximating Knapsack in Polynomial Time)**.** *Suppose $\mathbf{p}, \mathbf{w} \in \mathbb{Z}^n$ and D define a standard Knapsack problem with optimal solution $\mathbf{x}^*$. Given $\varepsilon > 0$, we can find $\bar{\mathbf{x}} \in \{0,1\}^n$ such that*

$$\mathbf{p}^\top \bar{\mathbf{x}} \geq (1 - \varepsilon)\, \mathbf{p}^\top \mathbf{x}^*$$

in polynomial time.

Proof. Firstly, for all $i \in [n]$, if $\mathbf{w}_i > D$ then item i can never be selected. So we can remove the item without changing the optimal solution. Therefore, without loss of generality, assume that all $\mathbf{w}_i \leq D$ for $i \in [n]$.

We will now construct a new rounded profits vector $\bar{\mathbf{p}}$:

$$\bar{\mathbf{p}}_i = \left\lceil \frac{\mathbf{p}_i}{\kappa} \right\rceil, \qquad \forall i \in [n]$$

where $\kappa = \frac{\mathbf{p}_{\max}\varepsilon}{n}$. Note that

$$\bar{\mathbf{p}}_i = \left\lceil \frac{\mathbf{p}_i n}{\mathbf{p}_{\max}\varepsilon} \right\rceil \leq \left\lceil \frac{n}{\varepsilon} \right\rceil$$

Furthermore, if $n \geq 2$ and $\varepsilon \leq \frac{1}{2}$, then $\bar{\mathbf{p}}_i \leq \frac{2n}{\varepsilon}$ for all $i \in [n]$.

We can run the algorithm from Solving Knapsack through Generating Table Alternative with $\bar{\mathbf{p}}$ instead of $\mathbf{p}$ to generate a solution $\bar{\mathbf{x}}$ in $\mathcal{O}\left(\frac{n^3}{\varepsilon}\right)$ time. Note that this is polynomial complexity with respect to all inputs. So we only need to show that $\bar{\mathbf{x}}$ is an ε-approximation. So consider:

$$\begin{aligned}
\mathbf{p}^\top \bar{\mathbf{x}} &= (\kappa\bar{\mathbf{p}} - k)^\top \bar{\mathbf{x}}, \text{ where } k \in [\kappa]^n \\
&= \kappa\bar{\mathbf{p}}^\top \bar{\mathbf{x}} - k^\top \bar{\mathbf{x}} \\
&\geq \kappa\bar{\mathbf{p}}^\top \mathbf{x}^* - k^\top \bar{\mathbf{x}} && (3.1) \\
&= \kappa\bar{\mathbf{p}}^\top \mathbf{x}^* - k^\top \mathbf{x}^* + k^\top \mathbf{x}^* - k^\top \bar{\mathbf{x}} \\
&= (\kappa\bar{\mathbf{p}} - k)^\top \mathbf{x}^* + k^\top (x^* - \bar{\mathbf{x}}) \\
&= \mathbf{p}^\top \mathbf{x}^* + k^\top (x^* - \bar{\mathbf{x}}) \\
&\geq \mathbf{p}^\top \mathbf{x}^* - \kappa n && (3.2) \\
&= \mathbf{p}^\top \mathbf{x}^* - \mathbf{p}_{\max}\varepsilon \\
&\geq (1-\varepsilon)\, \mathbf{p}^\top \mathbf{x}^* && (3.3)
\end{aligned}$$

where Inequality (3.1) follows from $\bar{\mathbf{x}}$ being the optimal solution for the Knapsack problem with profits $\bar{\mathbf{p}}$, Inequality (3.2) follows from $(\mathbf{x}^* - \bar{\mathbf{x}}) \in \{0, \pm 1\}^n$ and $\max k = \kappa$, and the Inequality (3.3) follows from $\mathbf{p}_{\max}$ being an obtainable solution given that $\mathbf{p}_i \leq D$ for all $i \in [n]$. Therefore, $\bar{\mathbf{x}}$ is an ε-approximation that is computable in polynomial time, as required. □

Corollary 3.1 (Approximating IPs via Rounding)**.** *All integer programs of the form*

$$\begin{aligned}
&\max\ \mathbf{c}^\top \mathbf{x} \\
&\quad s.t\ A\mathbf{x} \leq \mathbf{b}, x \in \{0,1\}^n
\end{aligned}$$

with $A \in \mathbb{N}^{m\times n}$ and $\mathbf{b} \in \mathbb{N}^n$ for which there is a pseudo-polynomial time algorithm can be can be approximated in polynomial time within $\varepsilon > 0$ accuracy.

Proof. Repeat the proof for the Approximating Knapsack in Polynomial Time Theorem, as it only dependent upon the problem form and existence of a pseudo-polynomial time algorithm. □

4 Encodings

Definition 4.1 (Size)**.** Given $a \in \mathbb{Z}$, the *Size* of a is $\operatorname{size}(a) = \log_2(\max\{|a|, 1\})$ (maximum is needed for the 0 case). The binary encoding length of an integer a is $1 + \lceil \operatorname{size}(a) \rceil$, where the first bit is the sign.

For $\mathbf{v} \in \mathbb{Z}^n$, $\operatorname{size}(\mathbf{v}) = \sum_{i\in[n]} \operatorname{size}(\mathbf{v}_i)$. Note that $\operatorname{size}(\mathbf{v}) \leq n \operatorname{size}(\|\mathbf{v}\|_\infty)$ for all $\mathbf{v} \in \mathbb{Z}^n$.

Example 4.1 (Simple Sizes). If $a = 5$, then $\text{size}(a) = \log_2(\max\{|5|, 1\}) \approx 2.3$. We can verify that $1 + \lceil \text{size}(a) \rceil = 4$ is the binary encoding length through manual conversion: $5 = \langle 1101 \rangle_2$ (recall the first bit is the sign bit).

If $b = -10$, then $\text{size}(b) = \log_2(\max\{|-10|, 1\}) \approx 3.3$. We can verify that $1 + \lceil \text{size}(a) \rceil = 5$ is the binary encoding length through manual conversion: $-10 = \langle 01010 \rangle_2$.

Example 4.2 (Polynomial Algorithm Producing Exponential Sized Output). Suppose an algorithm carries out a polynomial number (with respect to the size of the problem) of arithmetic operations. Will the output always be polynomial in size? Unfortunately, no.

Consider the following algorithm. Its only input is $\mathbf{a} \in \{0,1\}^n$, so problem size is $\Theta(n)$. It starts with $x = 2$, then $x \leftarrow x^2$ for each $\mathbf{a}_i = 1$, and finally outputs x: that is, beginning with 2, it will square the value for every active component of $\mathbf{a}$. If $\mathbf{a} = \mathbf{1}$, then the output is 2^{2^n}. So output $= 2^n$. So despite a polynomial number of arithmetic operations, this algorithm has an output with exponential size with respect to its input size.

Theorem 4.1 (Hadamard Bound). *For* $A \in \mathbb{R}^{n \times n}$, $\det(A) \leq \prod_{i \in [n]} \|\mathbf{a_i}\|_2$, *where* $A = \begin{pmatrix} \mathbf{a_1} & \dots & \mathbf{a_n} \end{pmatrix}$.

Proof. The Gram-Schmidt procedure takes as input $\{\mathbf{b_1}, \dots, \mathbf{b_n}\}$ which span $\mathbb{R}^n$ and outputs a linearly independent basis of $\mathbb{R}^n$ $\{\mathbf{b_1^*}, \dots, \mathbf{b_n^*}\}$, i.e. $\mathbf{b_i^*} \perp \mathbf{b_j^*}$ for $i \neq j$, such that

$$\text{span}(\{\mathbf{b_i} : i \in [k]\}) = \text{span}(\{\mathbf{b_i^*} : i \in [k]\})$$

for $k \in [n]$. The Gram-Schmidt process produces this special basis recursively:

$$\begin{aligned} \mathbf{b_1^*} &= \mathbf{b_1} \\ \mathbf{b_{i+1}^*} &= \mathbf{b_{i+1}} - \sum_{j \in [i]} \text{proj}_{\mathbf{b_j^*}}(\mathbf{b_{i+1}}) \end{aligned}$$

where $\text{proj}_{\mathbf{a}}(\mathbf{b}) = \frac{\langle \mathbf{a}, \mathbf{b} \rangle}{\langle \mathbf{b}, \mathbf{b} \rangle} \mathbf{b}$ is the orthogonal projection operator. Let $\mu_{i,j} = -\text{proj}_{\mathbf{b_j^*}}(\mathbf{b_{i+1}})$ for $j \in [i]$ and $i \in [n]$. If B and B^* are the matrices with rows $\mathbf{b_i}$ and $\mathbf{b_i^*}$, respectively, then Gram-Schmidt sets up the following matrix equation:

$$B = B^* \begin{pmatrix} 1 & \mu_{2,1} & \dots & \mu_{n,1} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mu_{n,n-2} \\ 0 & \dots & 0 & 1 \end{pmatrix}$$

i.e. the change of basis from $\mathbf{b_i^*}$'s to $\mathbf{b_i}$ is done through a upper-triangular matrix with diagonals 1 and the $\mu_{i,j}$'s above the diagonal. Let U be this upper triangular matrix. Now consider:

$$\begin{aligned} \det\left(B^T B\right) &= \det\left(\left(B^* T\right)^T\right)\left(B^* T\right) \\ \det(B)^2 &= \det(T)^2 \det\left(B^{*T} B^*\right) \\ &= 1^2 \times \det\left(B^{*T} B^*\right) \\ &= \det\left(\begin{pmatrix} \|\mathbf{b_1^*}\|_2^2 & & \star \\ & \ddots & \\ \star & & \|\mathbf{b_n^*}\|_2^2 \end{pmatrix}\right) \\ &\leq \prod_{i \in [n]} \|\mathbf{b_i^*}\|_2^2 \end{aligned}$$

And so, we have that:

$$\det(B) \leq \prod_{i \in [n]} \|\mathbf{b_i^*}\|_2$$

as required. $\square$

Corollary 4.1 (Size of Determinants)**.** *For $A \in \mathbb{Z}^{n \times n}$ with $\det(A) \neq 0$ and $\max\{|A_{i,j}| : i, j \in [n]\} = A_{\max} \leq \Delta$, then $\text{size}(\det(A)) \leq n(\log_2(\Delta) + \log_2(n))$.*

Proof. We apply the Hadamard Bound:

$$\begin{aligned}
\text{size}(\det(A)) &= \log_2(\max\{|\det(A)|, 2\}) \\
&\leq \log_2\left(\max\left\{\prod_{i \in [n]} \|\mathbf{a_i}\|_2, 2\right\}\right) \\
&\leq \log_2\left(\max\left\{\prod_{i \in [n]} \Delta\sqrt{n}, 2\right\}\right) \\
&= \log_2\left(\Delta^n n^{\frac{n}{2}}\right) \\
&\leq n(\log_2(\Delta) + \log_2(n))
\end{aligned}$$

as required. $\square$

Theorem 4.2 (Linear Equations Produce Polynomial Solutions)**.** *For $A \in \mathbb{Z}^{n \times n}$ with $\det(A) \neq 0$ and $\mathbf{b} \in \mathbb{Z}^n$ such that $A_{\max}, \mathbf{b}_{\max} \leq \Delta$, the output $\mathbf{x}^* \in \mathbb{Q}^n$ which solves the linear equation $A\mathbf{x}^* = \mathbf{b}$ has size polynomial with respect to the input size.*

Proof. We can use the adjunct of A, $\tilde{A}$, to represent $\mathbf{x}^*$:

$$\mathbf{x}^* = A^{-1}\mathbf{b} = \frac{\tilde{A}}{\det(A)}\mathbf{b}$$

Furthermore, we know that $\tilde{A}_{\max} = \max_{i,j} M_{i,j}$, where $M_{i,j}$ is the i, j-th minor of A. Let $\mathbf{a^j}$ represent $\mathbf{a}$ with the j-th component removed. Consider:

$$M_{i,j} \leq \prod_{i \in [n] \setminus \{i\}} \left\|\mathbf{a_i^j}\right\|_2 \leq \prod_{i \in [n] \setminus \{i\}} \|\mathbf{a_i}\|_2 \leq \prod_{i \in [n]} \|\mathbf{a_i}\|_2$$

where the first inequality follows from the Hadamard Bound, the second inequality follows from $\left\|\mathbf{a^j}\right\|_2 \leq \|\mathbf{a}\|_2$ for all $\mathbf{a} \in \mathbb{Z}^n$, and the final inequality follows from $\|\mathbf{a}\|_2 \geq 1$ for all $\mathbf{a} \in \mathbb{Z}^n$ except $\mathbf{a} = \mathbf{0}$ but no $\mathbf{a_i} = \mathbf{0}$ as $\det(A) \neq 0$. Therefore, we have that:

$$\tilde{A}_{\max} \leq \prod_{i \in [n]} \|\mathbf{a_i}\|_2 \leq \Delta^n n^{\frac{n}{2}}$$

where the second inequality follows from the same reasoning as done in the Size of Determinants Corollary. Therefore:

$$\left\|\tilde{A}\mathbf{b}\right\|_\infty \leq n\tilde{A}_{\max}\mathbf{b}_{\max} \leq n\left(\Delta^n n^{\frac{n}{2}}\right)\Delta = \Delta^{n+1} n^{\frac{n+2}{2}}$$

Given this, we can infer that $\text{size}\left(\tilde{A}\mathbf{b}\right) \leq n(n+1)(\log_2(\Delta) + \log_2(n))$. Using the Size of Determinants Corollary, we now know that the numerator and denominator for a representation of $\mathbf{x}^*$ both has size polynomial in the input, so $\mathbf{x}^*$ has a representation with size polynomial in the input. $\square$

Theorem 4.3 (Polynomial Time Determinant Algorithm)**.** *Given $A \in \mathbb{Z}^{n\times n}$ with $A_{\max} \le \Delta$, the determinant can be computed in polynomial time.*

Proof. Note this is merely a proof sketch.

Let $m = \Delta^n n^{\frac{n}{2}}$, which is larger than $|\det(A)|$ by the reasoning done in Corollary Size of Determinants. Let $p_1, \dots, p_k$ denote the smallest k primes such that $\prod_{i\in[k]} p_i \ge 2m$. For all $i \in [k]$, we can compute $\det(A)$ over $\mathbb{Z}_{p_i}$ in polynomial time using the Gaussian elimination. Given the isomorphism $\mathbb{Z}_{\prod_{i\in[k]} p_i} \cong \mathbb{Z}_{p_1} \times \dots \mathbb{Z}_{p_k}$ and $|\det(A)| \le m \le \frac{1}{2}\prod_{i\in[k]} p_i$, we use the Chinese Remainder theorem to find $\det(A)$.

Now we need to ensure that we can find $p_1, \dots, p_k$ in polynomial time. Let $\pi(l) = |\{p \in [l] : p \text{ is prime}\}|$, i.e number of primes less than or equal to l. The Prime Number Theorem implies that $\pi(l) \ge C\frac{l}{\log(l)}$ for some constant $C \in \mathbb{R}_+$, which we can weaken to $\pi(l) \ge C\sqrt{l}$. Additionally, the product of k primes must satisfy $\prod_{i\in[k]} p_i \ge 2^k$. Putting these bounds together, we only need to search up to $L = \left(\frac{\log_2(2m)}{C}\right)^2$ to find the required primes: using the weaker bound, there will be at least $\log_2(2m)$ primes less than L and the product of these primes will necessarily be at minimum $2m$. Therefore, we can use the Sieve of Eratosthenes to find these primes less than L in polynomial time with respect to L. Finally, we note that

$$L = \left(\frac{\log_2(2m)}{C}\right)^2 = \left(\frac{1 + \log_2\left(\Delta^n n^{\frac{n}{2}}\right)}{C}\right)^2 \le \left(\frac{1 + n\left(\log_2(\Delta) + \log_2(n)\right)}{C}\right)^2 \le \left(\frac{1 + n\Delta + n^2}{C}\right)^2$$

is polynomial in the input size. So overall, we can compute $\det(A)$ is polynomial time. $\square$

Definition 4.2 (Forms)**.** An IP of the form

$$\max \mathbf{c}^T\mathbf{x}$$
$$s.t\ A\mathbf{x} = \mathbf{b}, \mathbf{x} \in {\mathbb{Z}_+}^n$$

with $A \in \mathbb{Z}^{m\times n}$ and $\mathbf{b} \in \mathbb{Z}^m$ is in *Equation Standard Form*. On the other hand, an IP of the form

$$\max \mathbf{c}^T\mathbf{x}$$
$$s.t\ A\mathbf{x} \le \mathbf{b}, \mathbf{x} \in \mathbb{Z}^n$$

with $A \in \mathbb{Z}^{m\times n}$ and $\mathbf{b} \in \mathbb{Z}^m$ is in *Inequality Standard Form*. Note that Equality requires positive integer vectors $\mathbf{x} \in {\mathbb{Z}_+}^n$ where as Inequality allows all integer vectors $\mathbf{x} \in \mathbb{Z}^n$.

Proposition 4.1 (Equivalences between Standard Forms)**.** *An IP with n variables and m inequalities in Inequality Standard Form is equivalent to an IP with $2n+m$ variables and m equalities in Equality Standard Form.*

Proof. We can explicitly construct the equivalences:

$$\begin{array}{rcl} \max \mathbf{c}^\top \mathbf{x} & & \max \mathbf{c}^\top (\mathbf{x}_+ - \mathbf{x}_-) \\ s.t\ A\mathbf{x} \le \mathbf{b} & \equiv & s.t\ A(\mathbf{x}_+ - \mathbf{x}_-) \le \mathbf{b} \\ \mathbf{x} \in \mathbb{Z}^n & & \mathbf{x}_+, \mathbf{x}_- \in {\mathbb{Z}_+}^n \end{array}$$

$$\begin{array}{rcl} & & \max \mathbf{c}^\top (\mathbf{x}_+ - \mathbf{x}_-) \\ & \equiv & s.t\ A\mathbf{x}_+ - A\mathbf{x}_- + I_m\mathbf{y} = \mathbf{b} \\ & & \mathbf{x}_+, \mathbf{x}_-, \mathbf{y} \in {\mathbb{Z}_+}^n \end{array}$$

where $\mathbf{y}$ is introduced as a slack variable. □

Example 4.3 (Switching between Standard Forms). We present a problem in its equivalent Equality and Inequality Standard Forms:

$$\begin{array}{lcl} \max \begin{pmatrix} 2 \\ 1 \end{pmatrix}^\top \mathbf{x} & & \max \begin{pmatrix} 2 \\ 1 \\ -2 \\ -1 \end{pmatrix}^\top \begin{pmatrix} \mathbf{x}_+ \\ \mathbf{x}_- \end{pmatrix} \\ s.t \begin{pmatrix} 3 & 2 \\ 1 & 3 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 5 \\ 3 \end{pmatrix} & \equiv & s.t \begin{pmatrix} 3 & 2 & -3 & -2 & 1 & 0 \\ 1 & 3 & -1 & -3 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_+ \\ \mathbf{x}_- \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} 5 \\ 3 \end{pmatrix} \\ \mathbf{x} \in \mathbb{Z}^2 & & \mathbf{x}_+, \mathbf{x}_-, \mathbf{y} \in {\mathbb{Z}_+}^2 \end{array}$$

Note that the Equality form requires more variables to ensure the positivity and equalities.

Theorem 4.4 (Bound on Solution for Equality Standard Form)**.** *Suppose we have an IP in Equality Standard Form*

$$\begin{aligned} &\max \mathbf{c}^\top \mathbf{x} \\ &s.t\ A\mathbf{x} = \mathbf{b}, \mathbf{x} \in {\mathbb{Z}_+}^n \end{aligned}$$

where $A \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{b} \in \mathbb{Z}^m$ *such that* $A_{\max}, \mathbf{b}_{\max} \le \Delta \in \mathbb{R}_+$. *If the IP is feasible, then the optimal solution* $\mathbf{x}^* \in \mathbb{Z}^n$ *satisfies* $\|\mathbf{x}^*\|_\infty \le (m\Delta)^{5m}$ *and* $\operatorname{size}(\mathbf{x}^*) \in \mathcal{O}\left((nm\log(\Delta))^2\right)$.

Proof. TODO: Apparently future homework problem covers the first part.

Finally, given that $\|\mathbf{x}^*\|_\infty \le (m\Delta)^{5m}$, we can immediately infer that

$$\begin{aligned} \operatorname{size}(\mathbf{x}^*) &\le n \operatorname{size}(\|\mathbf{x}^*\|_\infty) \\ &= n \log_2\left(\max\left\{(m\Delta)^{5m}, 2\right\}\right) \\ &= 5mn \log_2(m\Delta) \\ &\le (mn\log_2(\Delta))^2 \end{aligned}$$

as required. □

Theorem 4.5 (HINTED BUT YET TO BE DISCUSSED)**.** *SOMETHING ALONG THESE LINES: Either let* $A \in \mathbb{Z}^{m\times n}$ *and* $\mathbf{b} \in \mathbb{Z}^m$, *xor* $\mathbf{a} \in \mathbb{Z}^{m'}$ *and* $\beta \in \mathbb{Z}$. *Given either option, we can construct the equivalence:*

$$\begin{array}{rcr} \max \mathbb{1}^\top \mathbf{x} & & \mathbb{1}^\top \mathbf{x} \\ s.t\ A\mathbf{x} = \mathbf{b} & \equiv & \mathbf{a}^\top \mathbf{x} = \beta \\ \mathbf{x} \in \{0,1\}^n & & \mathbf{x} \in \{0,1\}^n \end{array}$$

Example 4.4 (Optimality of Pseudo-Polynomial Algorithm for Knapsack). By Bound on Solution for Equality Standard Form Theorem, we have the following equivalences between problems:

$$\begin{aligned} \max\ & \mathbb{1}^\top \mathbf{x} \\ s.t\ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \in \{0,1\}^n \end{aligned} \quad \equiv \quad \begin{aligned} \max\ & \mathbb{1}^\top \mathbf{x} \\ s.t\ & \mathbf{a}^\top \mathbf{x} = \beta \\ & \mathbf{x} \in \{0,1\}^n \end{aligned}$$

$$\equiv \quad \begin{aligned} \max\ & \mathbb{1}^\top \mathbf{x} \\ s.t\ & \mathbf{a}^\top \mathbf{x} \leq \beta \\ & \mathbf{x} \in \{0,1\}^n \end{aligned}$$

for some pairs of $A \in \mathbb{Z}^{m\times n}, \mathbf{b} \in \mathbb{Z}^m$ and $\mathbf{a} \in \mathbb{Z}^{m'}$ and $\beta \in \mathbb{Z}$. The final problem is a Knapsack problem with $\mathbf{p} = \mathbf{w} = \mathbf{a}$ and $D = \beta$. Therefore, if we could solve Knapsack in polynomial time (not pseudo-polynomial time), then we could the first general feasibility problems in polynomial time as well. This is unlikely, so this suggests that Knapsack is pseudo-polynomial at minimal.