COMPLEXITY OF THE SIMPLEX ALGORITHM AND POLYNOMIAL-TIME ALGORITHMS

Béatrice Byukusenge

Linkping University

January 12, 2017

- **Discuss** fundamental computational complexity issues for algorithms for solving linear programming problems.
- $\boldsymbol{f(n)}$ denotes " *the total number of elementary operations required by the algorithm to solve the problem of size* **n**".
- $\boldsymbol{f(n) = \mathcal{O}(n^k) \Leftrightarrow \exists \tau > 0 : f(n) \leq \tau n^k}$: **Polynomial-time** (theoretically efficient).
- $\boldsymbol{f(n) = \mathcal{O}(k^n) \Leftrightarrow \exists \tau > 0 : f(n) \leq \tau k^n}$: **exponential growth** (bad!). e.g.: *simplex algorithm*.
- There exist theoretically efficient algorithms for LP problems:
  - Khachian (no practical value).
  - Karmarkar (promising).

Consider the LP optimization problem:

$$\begin{aligned} \text{minimize} \quad & z(x) = cx \\ \text{s. to} \quad & Ax = b \\ & \mathbb{R}^n \ni x \geq 0 \end{aligned}$$

Data: $A \in \mathbb{R}^{m \times n}$; $c \in \mathbb{R}^n$; $b \in \mathbb{R}^m$ with $m, n \geq 2$.

- **size:** (m, n, L), where L is the **input length:** the number of binary bits required to record all the data of the problem (here $\log = \log_2$):

$$\begin{aligned} L = & \left\{1 + \left\lceil \log(1+m) \right\rceil\right\} + \left\{1 + \left\lceil \log(1+n) \right\rceil\right\} \\ & + \sum_j \left\{1 + \left\lceil \log(1+|c_j|) \right\rceil\right\} + \sum_i \sum_j \left\{1 + \left\lceil \log(1+|a_{ij}|) \right\rceil\right\} \\ & + \sum_i \left\{1 + \left\lceil \log(1+|b_i|) \right\rceil\right\}. \end{aligned}$$

We are only required to determine a function $g(m, n, L)$ in terms of (m, n, L) such that for some sufficiently large constant $\tau > 0$, we have

- $\boldsymbol{f(n, m, L)} \leq \tau g(m, n, L)$. i.e., $\boldsymbol{\mathcal{O}(g(m, n, L))}$.

Example: For algorithm actually involving a maximum of $\boldsymbol{f(n, m)} = 6m^2 n + 15mn + 12m$ is $\boldsymbol{\mathcal{O}\left(m^2, n\right)}$.

We are only required to determine a function $g(m,n,L)$ in terms of (m,n,L) such that for some sufficiently large constant $\tau > 0$, we have

- $\boldsymbol{f(n,m,L)} \leq \tau g(m,n,L)$. i.e., $\boldsymbol{\mathcal{O}(g(m,n,L))}$.

Example: For algorithm actually involving a maximum of $\boldsymbol{f(n,m)} = 6m^2n + 15mn + 12m$ is $\boldsymbol{\mathcal{O}\left(m^2,n\right)}$.

Optimization Problem

$$\begin{aligned} \text{maximize} \quad & z(x) = cx \\ \text{s. to} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

Decision Problem

Given $\boldsymbol{c}$, $\boldsymbol{b}$ and $\boldsymbol{A}$ (of the appropriate dimensions) and given rational number K, does there exist a rational vector $\boldsymbol{x}$ such that $\boldsymbol{Ax = b}$, $\boldsymbol{x} \geq \boldsymbol{0}$, and $\boldsymbol{cx} \leq K$?

We are only required to determine a function $g(m, n, L)$ in terms of (m, n, L) such that for some sufficiently large constant $\tau > 0$, we have

- $\boldsymbol{f(n, m, L)} \leq \tau g(m, n, L)$. i.e., $\boldsymbol{\mathcal{O}(g(m, n, L))}$.

Example: For algorithm actually involving a maximum of $\boldsymbol{f(n, m)} = 6m^2n + 15mn + 12m$ is $\mathcal{O}\left(\boldsymbol{m^2, n}\right)$.

Optimization Problem

$$\begin{aligned} \text{maximize} \quad & z(x) = cx \\ \text{s. to} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

Decision Problem

Given $\boldsymbol{c}$, $\boldsymbol{b}$ and $\boldsymbol{A}$ (of the appropriate dimensions) and given rational number K, does there exist a rational vector $\boldsymbol{x}$ such that $\boldsymbol{Ax = b}$, $\boldsymbol{x} \geq \boldsymbol{0}$, and $\boldsymbol{cx} \leq K$?

Theorem

polynomial-time algorithms for optimization problems $\Leftrightarrow$ those for decision problems.

Dantzig introduces the **simplex algorithm**.

- **intuition-based reaction:** the algorithm would not prove to be very efficient.
- **surprisingly:** in practice, this method performes exceedingly well.

Theoretically, the fact is that the algorithm is entrapped in the potentially combinatorial aspect of having to examine up to (for $n > m$):

$$\binom{n}{m} > \left(\frac{n}{m}\right)^m \text{ vertices.}$$

- Hence the plausibility of a potential **exponential order of effort for some problems**.

Example: 1971 Klee-Minty problems: Feasible region is a suitable distortion of the n-dimensional hypercube in $\mathbb{R}^n$ which has 2^n vertices.

Problem $(\varepsilon \in (0, 1/2))$

$$\begin{array}{ll} \text{Maximize} & x_n \\ \text{s. to} & 0 \leq x_1 \leq 1 \\ & \varepsilon x_{j-1} \leq x_j \leq 1 - \varepsilon x_{j-1} \\ & (\text{for } j = 2, \ldots, n) \\ & x_j \geq 0, \ j = 1, \ldots, n. \end{array}$$

Transformedd Problem $(\theta = 1/\varepsilon)$

$$\begin{array}{ll} \text{Maximize} & \sum_{j=1}^{n} y_j \\ \text{s. to} & y_1 \leq 1 \\ & y_j + 2 \sum_{k=1}^{j-1} y_k \leq \theta^{j-1} \\ & (\text{for } j = 2, \ldots, n) \\ & y_j \geq 0, \ j = 1, \ldots, n. \end{array}$$

where $y_1 = x_1, \ y_j = \left(x_j - \varepsilon x_{j-1}\right)/\varepsilon^{j-1}$ for $j = 2, \ldots, n$.

- $2^n - 1$ iterations to visit all the 2^n vertices.

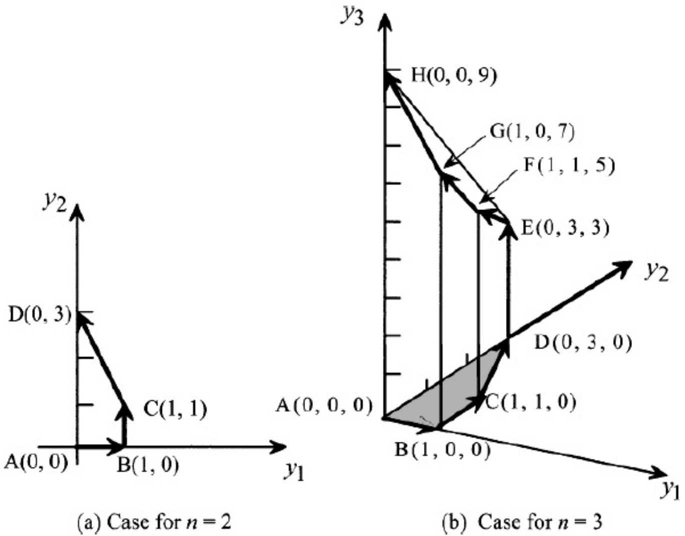

Figure 8.1. Illustration of the Klee–Minty type polytopes for n=2 and n=3.

In 1984 Karmarkar (AT&T Bell Laboratories) proposed a new **polynomial-time** algorithm for LP problems. This algorithm addresses LP problems of the following form:

$$\begin{aligned} \text{Minimize} \quad & z = cx \\ \text{s. to} \quad & Ax = 0 \\ & \mathbf{1}x = 1 \qquad \text{(LP-K)} \\ & x \geq 0 \end{aligned}$$

where $A \in \mathbb{R}^{m \times n}$, with $m, n \geq 2$, c, A integers and $\mathbf{1}$ is a row vector of n ones with the following two assumptions:

- (A_1): $x_0 = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)^T$ is feasible.
- (A_2): $z^* = 0$.

Any general LP problem can be (*polynomially*) cast in this form through the use of **artificial variables**, an **artificial bounding constraint**, and through **variable redefinitions**.

- **Remark:** Under assumptions (A_1) and (A_1), Problem $(LP-K)$ is **feasible** and **bounded**, and hence, has an **optimum**.

- **Feasible region:** $K = \{Ax = 0\} \cap \{S_x \{x : \mathbf{1}x = 1, \ x \geq 0\}\}$

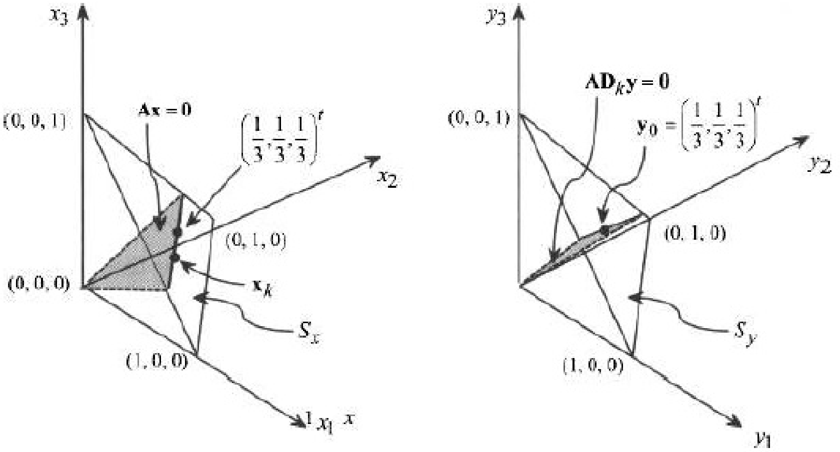

Figure 8.2. Projective transformation of the feasible region.

Summary of Karmarkar's Algorithm

- **INITIALIZATION**

Compute $r = 1/\sqrt{n(n-1)}$, $L = \left\lceil 1 + \log\left(1 + |c_{j\,\max}|\right) + \log\left(|\det_{\max}|\right)\right\rceil$, and select $\alpha = (n-1)/3n$. Let $\mathbf{x}_0 = (1/n, \ldots, 1/n)^t$ and put $k = 0$.

MAIN STEP

If $\mathbf{c}\mathbf{x}_k < 2^{-L}$, use the optimal rounding routine to determine an optimal solution, and stop. (Practically, since 2^{-L} may be very small, one may terminate when $\mathbf{c}\mathbf{x}_k$ is less than some other desired tolerance.) Otherwise, define

$$\mathbf{D}_k = \operatorname{diag}\{x_{k1}, \ldots, x_{kn}\}, \qquad \mathbf{y}_0 = \left(\frac{1}{n}, \ldots, \frac{1}{n}\right)^t,$$

$$\mathbf{P} = \begin{bmatrix} \mathbf{A}\mathbf{D}_k \\ \mathbf{1} \end{bmatrix} \qquad \text{and} \qquad \overline{\mathbf{c}} = \mathbf{c}\mathbf{D}_k$$

and compute

$$\mathbf{y}_{\text{new}} = \mathbf{y}_0 - \alpha r \frac{\mathbf{c}_p}{\|\mathbf{c}_p\|}, \qquad \text{where } \mathbf{c}_p = \left[\mathbf{I} - \mathbf{P}^t(\mathbf{P}\mathbf{P}^t)^{-1}\mathbf{P}\right]\overline{\mathbf{c}}^t.$$

Hence, obtain $\mathbf{x}_{k+1} = (\mathbf{D}_k \mathbf{y}_{\text{new}})/(\mathbf{1}\mathbf{D}_k \mathbf{y}_{\text{new}})$. Increment k by one and repeat the Main Step.

- **OPTIMAL ROUNDING ROUTINE**

Starting with $\mathbf{x}_k$, determine an extreme point solution $\bar{\mathbf{x}}$ for Problem (8.4) with $\mathbf{c}\bar{\mathbf{x}} \leq \mathbf{c}\mathbf{x}_k < 2^{-L}$, using the earlier *purification scheme*. Terminate with $\bar{\mathbf{x}}$ as an optimal solution to Problem (8.4).

Thank you for your attention!