Linear Programming: Chapter 4
Efficiency

Robert J. Vanderbei

October 17, 2007

Operations Research and Financial Engineering
Princeton University
Princeton, NJ 08544
http://www.princeton.edu/∼rvdb

Efficiency

Question:

Given a problem of a certain size, how long will it take to solve it?

Two Kinds of Answers:

- *Average Case.* How long for a *typical* problem.
- *Worst Case.* How long for the *hardest* problem.

Average Case.

- Mathematically difficult.
- Empirical studies.

Worst Case.

- Mathematically tractible.
- Limited value.

Measures

Measures of Size

- Number of constraints m and/or number of variables n.
- Number of data elements, mn.
- Number of nonzero data elements.
- Size, in bytes, of AMPL formulation (model+data).

Measuring Time

Three factors:

- Number of iterations.
- Arithmetic operations per iteration.
- Time per arithmetic operation (depends on hardware).

Klee–Minty Problem (1972)

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{j=1}^{n} 2^{n-j} x_j & \\ \text{subject to} & \displaystyle 2\sum_{j=1}^{i-1} 2^{i-j} x_j + x_i \leq 100^{i-1} & i = 1, 2, \ldots, n \\ & x_j \geq 0 & j = 1, 2, \ldots, n. \end{array}$$

Example $n = 3$:

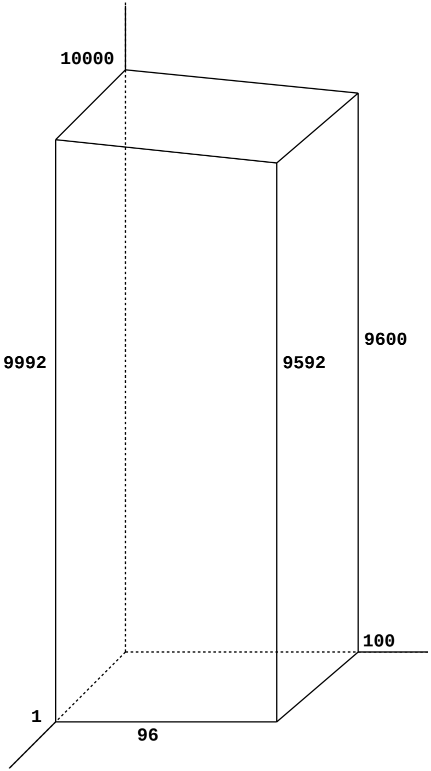
$$\begin{array}{rrcrcrcr} \text{maximize} & 4x_1 & + & 2x_2 & + & x_3 & & \\ \text{subj. to} & x_1 & & & & & \leq & 1 \\ & 4x_1 & + & x_2 & & & \leq & 100 \\ & 8x_1 & + & 4x_2 & + & x_3 & \leq & 10000 \\ & & & x_1, & x_2, & x_3 & \geq & 0. \end{array}$$

A Distorted Cube

Constraints represent a “minor” distortion to an n-dimensional hypercube:

$$\begin{aligned} 0 \leq \ & x_1 & \leq 1 \\ 0 \leq \ & x_2 & \leq 100 \\ & \vdots & \\ 0 \leq \ & x_n & \leq 100^{n-1}. \end{aligned}$$

Case $n = 3$:



Analysis

Replace

$$1, 100, 10000, \ldots,$$

with

$$1 = b_1 \ll b_2 \ll b_3 \ll \ldots.$$

Then, make following replacements to rhs:

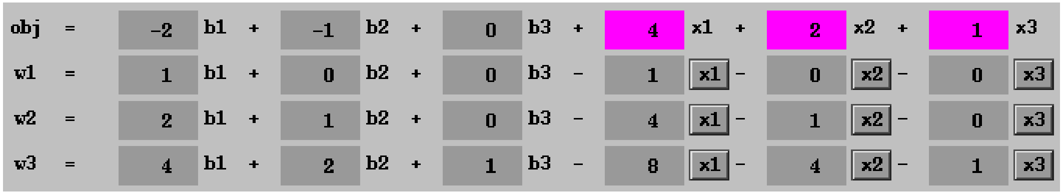
$$\begin{array}{rcl} b_1 & \longrightarrow & b_1 \\ b_2 & \longrightarrow & 2b_1 + b_2 \\ b_3 & \longrightarrow & 4b_1 + 2b_2 + b_3 \\ b_4 & \longrightarrow & 8b_1 + 4b_2 + 2b_3 + b_4 \\ & \vdots & \end{array}$$

Hardly a change!

Make a similar constant adjustment to objective function.

Look at the pivot tool version...

Case $n = 3$:



obj	=	-2	b1	+	-1	b2	+	0	b3	+	4	x1	+	2	x2	+	1	x3
w1	=	1	b1	+	0	b2	+	0	b3	–	1	x1	–	0	x2	–	0	x3
w2	=	2	b1	+	1	b2	+	0	b3	–	4	x1	–	1	x2	–	0	x3
w3	=	4	b1	+	2	b2	+	1	b3	–	8	x1	–	4	x2	–	1	x3

Now, watch the pivots...

Exponential

Klee–Minty problem shows that:

> Largest-coefficient rule can take $2^n - 1$ pivots to solve a problem in n variables and constraints (thereby visiting all 2^n vertices of the distorted cube).

For $n = 70$,

$$2^n = 1.2 \times 10^{21}.$$

At 1000 iterations per second, this problem will take 40 billion years to solve. The age of the universe is estimated at 13.7 billion years.

Yet, problems with 10,000 to 100,000 variables are solved routinely every day.

Worst case analysis is just that: worst case.

Complexity

n	n^2	n^3	2^n
1	1	1	2
2	4	8	4
3	9	27	8
4	16	64	16
5	25	125	32
6	36	216	64
7	49	343	128
8	64	512	256
9	81	729	512
10	100	1000	1024
12	144	1728	4096
14	196	2744	16384
16	256	4096	65536
18	324	5832	262144
20	400	8000	1048576
22	484	10648	4194304
24	576	13824	16777216
26	676	17576	67108864
28	784	21952	268435456
30	900	27000	1073741824

Sorting: $= n \log n$
Matrix times vector: n^2
Matrix times matrix: n^3
Matrix inversion: n^3
Simplex Method:

- Worst case: $n^2 2^n$ operations.
- Average case: n^3 operations.
- Open question:

  > Does there exist a variant of the simplex method whose worst case performance is polynomial?

Linear Programming:

- *Theorem:* There exists an algorithm whose worst case performance is $n^{3.5}$ operations.