Computational Geometry

Linear Programming

or

Profit Maximization

Lecture #4

[Comp. Geom A&A : Chapter 4]
Maximizing Profit

You are the boss of a small company that produces two products, $P_1$ and $P_2$. If you produce $x_1$ units of $P_1$ and $x_2$ units of $P_2$, your profit in € is

$$G(x_1, x_2) = 300x_1 + 500x_2$$

Your production runs on three machines $M_A$, $M_B$, and $M_C$ with the following capacities:

$$M_A: \quad 4x_1 + 11x_2 \leq 880$$
$$M_B: \quad x_1 + x_2 \leq 150$$
$$M_C: \quad x_2 \leq 60$$

Which choice of $(x_1, x_2)$ maximizes your profit?
The Answer

**linear constraints:**

\[
\begin{align*}
M_A &: 4x_1 + 11x_2 \leq 880 \\
M_B &: x_1 + x_2 \leq 150 \\
M_C &: x_2 \leq 60
\end{align*}
\]

\[
Ax \leq b
\]

\[
x \geq 0
\]

**linear objective fct.:**

\[
G(x_1, x_2) = 300x_1 + 500x_2 = (300, 500)(x_1)
\]

\[
G(110, 40) = 53,000
\]

\[
= \text{maximal value of objective fct. given constraints}
\]

\[
= \max\{c^T x \mid Ax \leq b, x \geq 0\}
\]

"iso-profit line" (orthogonal to \((300, 500)\))
Definition and Known Algorithms

Given a set $H$ of $n$ halfspaces in $\mathbb{R}^d$ and a direction $c$, find a point $x \in \bigcap H$ such that $cx$ is maximum (or minimum).

Many algorithms known, e.g.:
- Simplex [Dantzig ’47]
- Ellipsoid method [Khatchiyan ’79]
- Inner-point method [Karmakar’ 84]

Good for instances where $n$ and $d$ are large.

We consider $d = 2$.

VERY important problem, for example, in Operations Research.

[“Book” application: casting] $\bigcap H$ bounded.

\[ \bigcap H = \emptyset \]
\[ \bigcap H \text{ unbd. in dir. } c \]
set of optima: segment vs. point
First Approach

• compute $\bigcap H$ explicitly
• walk along $\partial (\bigcap H)$ to find a vertex $x$ with $c_x$ maximum

IntersectHalfplanes($H$)
if $|H| = 1$ then
    $C \leftarrow h$, where $\{h\} = H$
else
    split $H$ into sets $H_1$ and $H_2$ with $|H_1|, |H_2| \approx |H|/2$
    $C_1 \leftarrow \text{IntersectHalfplanes}(H_1)$
    $C_2 \leftarrow \text{IntersectHalfplanes}(H_2)$
    $C \leftarrow \text{IntersectConvexRegions}(C_1, C_2)$
return $C$

Running time: $T_{IH}(n) = 2T_{IH}(n/2) + T_{ICR}(n)$
Intersecting Convex Regions

Any ideas?

Use sweep-line algorithm for map overlay (line-segment intersections)!

Running time $T_{ICR}(n) = O((n + I) \log n)$,

where $I = \# \text{intersection points.}$

here: $I \leq n$

Running time $T_{IH}(n) = 2T_{IH}(n/2) + T_{ICR}(n)$

$\leq 2T_{IH}(n/2) + O(n \log n)$

$\in O(n \log^2 n)$

Better ideas?

Use specialized algorithm for intersecting convex regions/polyg.
Intersecting Convex Regions Faster

**Theorem.** The intersection of two convex polygonal regions can be computed in linear time.

**Corollary.** The intersection of $n$ half planes can be computed in $O(n \log n)$ time.

Can we do better?
A Small Trick: Make Solution Unique

- Add two bounding halfplanes $m_1$ and $m_2$

$$m_1 = \begin{cases} 
  x \leq M & \text{if } c_x > 0, \\
  x \geq M & \text{otherwise,}
\end{cases}$$

$$m_2 = \begin{cases} 
  y \leq M & \text{if } c_y > 0, \\
  y \geq M & \text{otherwise.}
\end{cases}$$

- Take the lexicographically largest solution.

⇒ Set of solutions is either empty or a uniquely defined point.
Incremental Approach

Idea: Don’t compute $\bigcap H$, but just one (optimal) point!

\[
2\text{dBoundedLP}(H, c, m_1, m_2)
\]

- Compute random permutation of $H$
- $H_0 = \{m_1, m_2\}; \ C_0 \leftarrow m_1 \cap m_2$
- $v_0 \leftarrow$ unique optimal vertex of $C_0$ wrt obj.

\[
\text{for } i \leftarrow 1 \text{ to } n \text{ do}
\]

\[
H_i = H_{i-1} \cup \{h_i\}; \ C_i = C_{i-1} \cap h_i
\]

if $v_{i-1} \in h_i$ then
\[
\quad v_i \leftarrow v_{i-1}
\]
else
\[
\quad v_i \leftarrow 1\text{dBoundedLP}(\pi_{\partial h_i}(H_{i-1}), \pi_{\partial h_i}(c))
\]

if $v_i = \text{nil}$ then
\[
\quad \text{return nil}
\]

return $v_n$

\[
\text{w-c running time:}
\]

\[
T(n) = \sum_{i=1}^{n} O(i) = O(n^2)
\]
Result

**Theorem.** The 2d bounded LP problem can be solved in $O(n)$ expected time.

**Proof.** Let $X_i = \begin{cases} 1 & \text{if } v_{i-1} \not\in h_i, \\ 0 & \text{else}. \end{cases}$ (indicator random variable).

Then the expected running time is

$$E[T_{2d}(n)] = E\left[\sum_{i=1}^{n}(1 - X_i) \cdot O(1) + X_i \cdot O(i)\right]$$

$$= O(n) + \sum E[X_i] \cdot O(i)$$

$$= O(n) + \sum \Pr[X_i = 1] \cdot O(i) = O(n).$$

We fix the $i$ random halfplanes in $H_i$. This fixes $C_i$.

$\Pr[X_i = 1]$ = probability that the optimal solution changes when $h_i$ is added to $C_{i-1}$.

$\Pr[X_i = 1]$ = probability that the optimal solution changes when $h_i$ is removed from $C_i$.

$\leq 2/i$. This is independent of the choice of $H_i$.

**Proof technique:** Backward analysis!