Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Chair for
INFORMATICS I
Efficient Algorithms and
Knowledge-Based Systems

Institute for Informatics

Computational Geometry

Winter term 2016/17

Linear Programming
or
Profit Maximization

Lecture #4

Joachim Spoerhase *Chair for Computer Science I*

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:

$$\begin{aligned} M_A: &\quad 4x_1 + 11x_2 \leq 880 \\ M_B: &\quad x_1 + x_2 \leq 150 \\ M_C: &\quad x_2 \leq 60 \end{aligned}$$

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

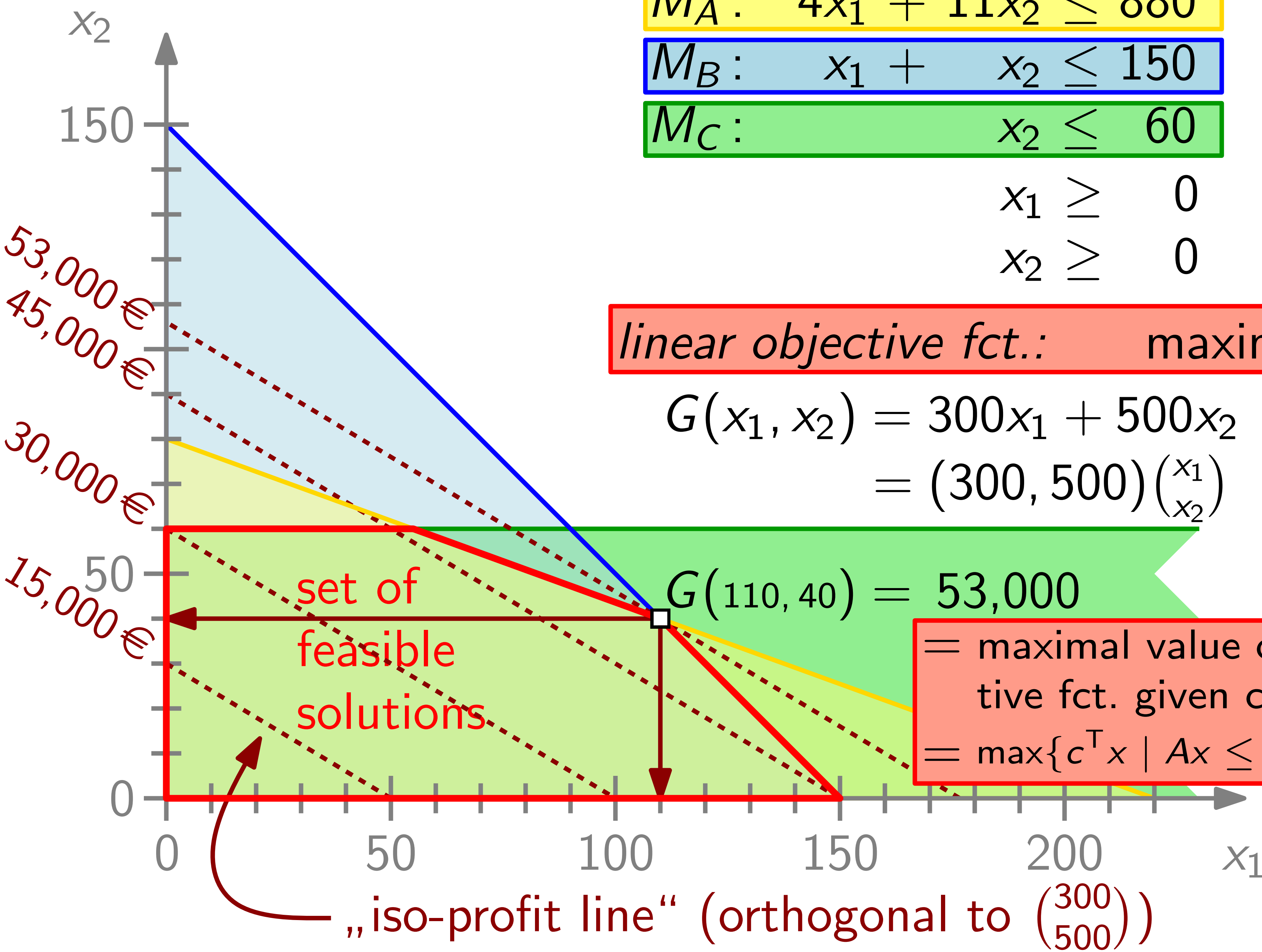
The Answer

linear constraints:

$$
\begin{aligned}
M_A: &\quad 4x_1 + 11x_2 \le 880 \\
M_B: &\quad x_1 + x_2 \le 150 \\
M_C: &\quad x_2 \le 60 \\
&\quad x_1 \ge 0 \\
&\quad x_2 \ge 0
\end{aligned}
\qquad
\begin{aligned}
Ax &\le b \\
x &\ge 0
\end{aligned}
$$

linear objective fct.: maximize $c^{\mathsf{T}}x$

$$
\begin{aligned}
G(x_1, x_2) &= 300x_1 + 500x_2 \\
&= (300, 500)\binom{x_1}{x_2}
\end{aligned}
$$



$G(110, 40) = 53{,}000$

$=$ maximal value of objective fct. given constraints

$= \max\{c^{\mathsf{T}}x \mid Ax \le b, x \ge 0\}$

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).

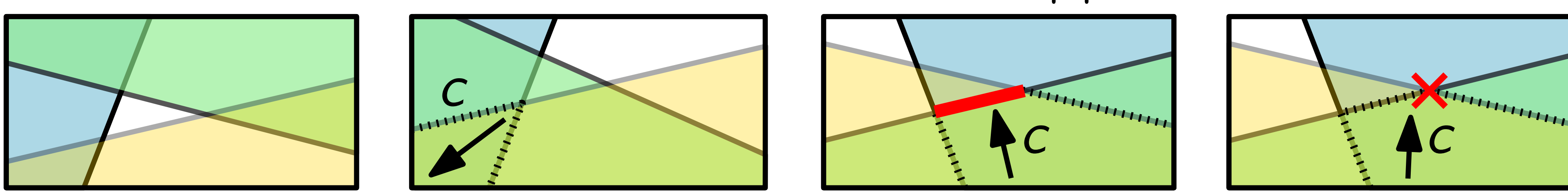
Many algorithms known, e.g.:
- Simplex [Dantzig '47]
- Ellipsoid method [Khatchiyan '79]
- Inner-point method [Karmakar' 84]

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.



$\bigcap H = \emptyset$ $\bigcap H$ unbnd. 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 cx maximum

IntersectHalfplanes(H)

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

How??

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

Intersecting Convex Regions

Any ideas?

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

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

where $I = \#$ intersection points.

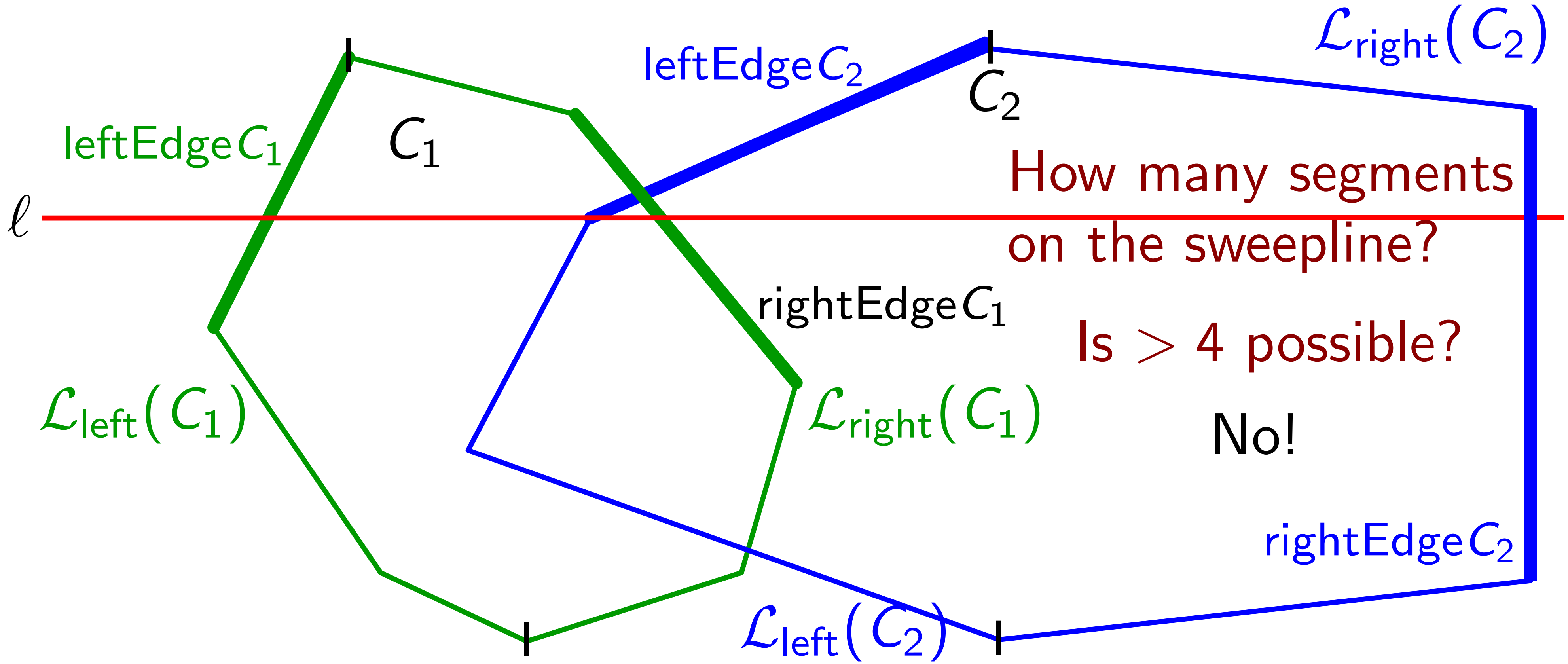
here: $I \leq n$

$$\begin{aligned} \text{Running time} \quad T_{\mathrm{IH}}(n) &= 2T_{\mathrm{IH}}(n/2) + T_{\mathrm{ICR}}(n) \\ &\leq 2T_{\mathrm{IH}}(n/2) + O(n \log n) \\ &\in O(n \log^2 n) \end{aligned}$$

Better ideas?

Use specialized algorithm for intersecting *convex* regions/polyg.

Intersecting Convex Regions Faster



How many segments on the sweepline?

Is > 4 possible?

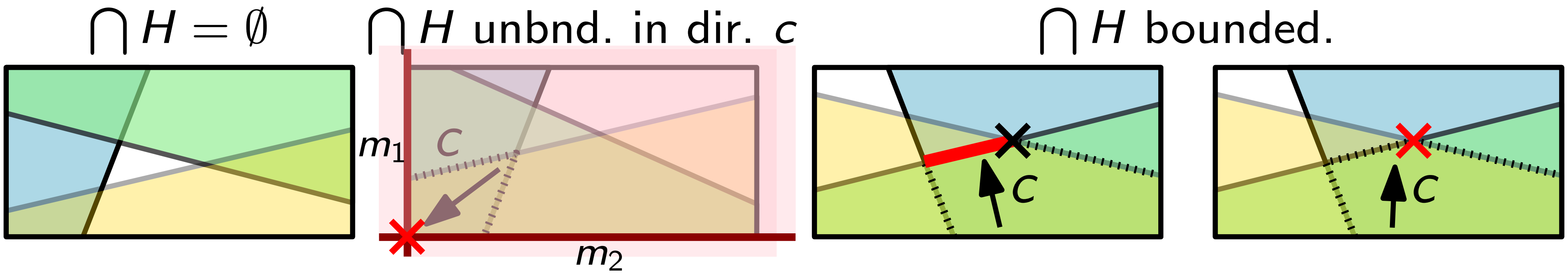
No!

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

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

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

- Take the lexicographically largest solution.

$\Rightarrow$ Set of solutions is either empty or a uniquely defined point.

Incremental Approach

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

Randomized

C_i = convex hull of H_i

2dBoundedLP(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.
for $i \leftarrow 1$ **to** n **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**
    $v_i \leftarrow v_{i-1}$ $O(1)$
  else
    $v_i \leftarrow$ 1dBoundedLP$(\pi_{\partial h_i}(H_{i-1}), \pi_{\partial h_i}(c))$ $O(i)$
    if $v_i =$ nil **then**
      return nil
return v_n

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 $X_i = \left\{\begin{matrix} 1 & \text{if } v_{i-1} \notin h_i, \\ 0 & \text{else.} \end{matrix}\right\}$ (indicator random variable).

Then the expected running time is

$$\begin{aligned} \mathbf{E}[T_{2d}(n)] &= \mathbf{E}\left[\textstyle\sum_{i=1}^{n} (1 - X_i) \cdot O(1) + X_i \cdot O(i)\right] \\ &= O(n) + \textstyle\sum \mathbf{E}[X_i] \cdot O(i) \\ &= O(n) + \textstyle\sum \mathbf{Pr}[X_i = 1] \cdot O(i) \quad = O(n). \end{aligned}$$

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

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

Proof technique: *Backward analysis!*

$=$ 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. □