Lecture 4: Outline

1. Existence of extreme points.
2. Optimality of extreme points.
3. On the running time of algorithms and some complexity theory.
Existence of Extreme Points

A polyhedron $P \subset \mathbb{R}^n$ contains a line if $\exists \ x \in P$ and $d \in \mathbb{R}^n$ $(d \neq 0)$ s.t. $x + \lambda d \in P$, $\forall \lambda \in \mathbb{R}$.

**Theorem**

Let $P = \{ x \in \mathbb{R}^n \mid a_i^t x \geq b_i, \ i = 1, \ldots, m \} \neq \emptyset$. Then

$P$ has a BFS $\iff P$ contains no line

Special cases:
- Bounded Polyhedra do not contain a line $\Rightarrow$ have a BFS.
- Standard form polyhedra are in the positive orthant $\Rightarrow$ do not contain a line $\Rightarrow$ have a BFS.

Optimality of extreme points

Consider the LP of minimizing $c^t x$ over a polyhedron $P \neq \emptyset$. WLOG assume $P$ has a bfs (otherwise can write LP in standard form and then $P$ will have a bfs).

**Theorem** *(The central LP thm.)* Either

- Optimal cost $= -\infty$, or
- \ exists a bfs which is optimal.

**Proof:** Start from an arbitrary $x \in P$. We can keep decreasing the cost by moving from point to point until we hit a bfs (otherwise the cost is $-\infty$).
Remarks

- Previous thm. says that to find the optimum it suffices to consider only extreme points. This is what the simplex method does. Moving from bfs to bfs until it finds an optimum.

- In LP it is holds that local minima are also global minima, which is a far more general property. We will see that it holds in convex programming (optimization of convex function over a convex set). LP is a special case of convex programming.

Notation for the Running time of algorithms

Let \( f, g : \mathbb{R}_+ \rightarrow \mathbb{R}_+ \)

**Definition**

\[ f(n) = O(g(n)) \] if \( \exists n_0, c \geq 0 \) such that \( f(n) \leq cg(n) \ \forall n \geq n_0 \).

**Definition**

\[ f(n) = \Omega(g(n)) \] if \( \exists n_0, c \geq 0 \) such that \( f(n) \geq cg(n) \ \forall n \geq n_0 \).

**Definition**

\[ f(n) = \Theta(g(n)) \] if both \( f(n) = O(g(n)) \) and \( f(n) = \Omega(g(n)) \) hold.
Definition (Reductions) Let $\Pi_1, \Pi_2$ two recognition problems. We say $\Pi_1$ transforms or reduces to $\Pi_2$ in polynomial time if there exists a poly-time algorithm which given an instance $I_1$ of $\Pi_1$ outputs an instance $I_2$ of $\Pi_2$ with the property: $I_1$ is a YES iff $I_2$ is a YES.

Definition (NP-hard) A problem is NP-hard if ZOIP can be transformed to it in poly-time.

Definition (Belongs to NP) A problem belongs to NP if it can be transformed to ZOIP in poly-time.

Definition (NP-complete) A problem is NP-complete if it belongs to NP and is also NP-hard.