CSL 356: Analysis and Design of Algorithms

Ragesh Jaiswal

CSE, IIT Delhi

Topics

- Greedy Algorithms
- Divide and Conquer
- Dynamic Programming
- Network Flow
- Computational intractability
- Other topics: Linear Programming

Linear Programming

Linear Programming: Introduction

- A large class of optimization problems in which the constraints and optimization criterion are linear functions.
- A *Linear Programming(**LP**)* problem consists of assigning real values to variables such that these variables
  1. (**Linear constraints**) satisfy a set of *linear* equalities or inequalities, and
  2. (**Objective function**) maximize or minimize a given *linear* objective function.

Linear Programming: Introduction

- Example: A cottage industry makes two kinds of products P_1 and P_2. The daily demand for P_1 is 100 and the daily demand for P_2 is 200. The total amount of items that the industry can produce in a day is 250. The industry makes profit of $Rs. 1$ per unit item of type P_1 and $Rs. 5$ per unit item of type P_2. How many items of P_1 and P_2 should the industry produce to make maximum amount of profit?
- Let x_1 be a variable denoting the amount of P_1 items produced by the industry and x_2 the mount of P_2 items.
- The goal is to maximize the *linear objective function:*

$$1 \cdot x_1 + 5 \cdot x_2$$

under the *linear constraints*:

$x_1 \geq 0, x_2 \geq 0, x_1 \leq 100, x_2 \leq 200, x_1 + x_2 \leq 250$

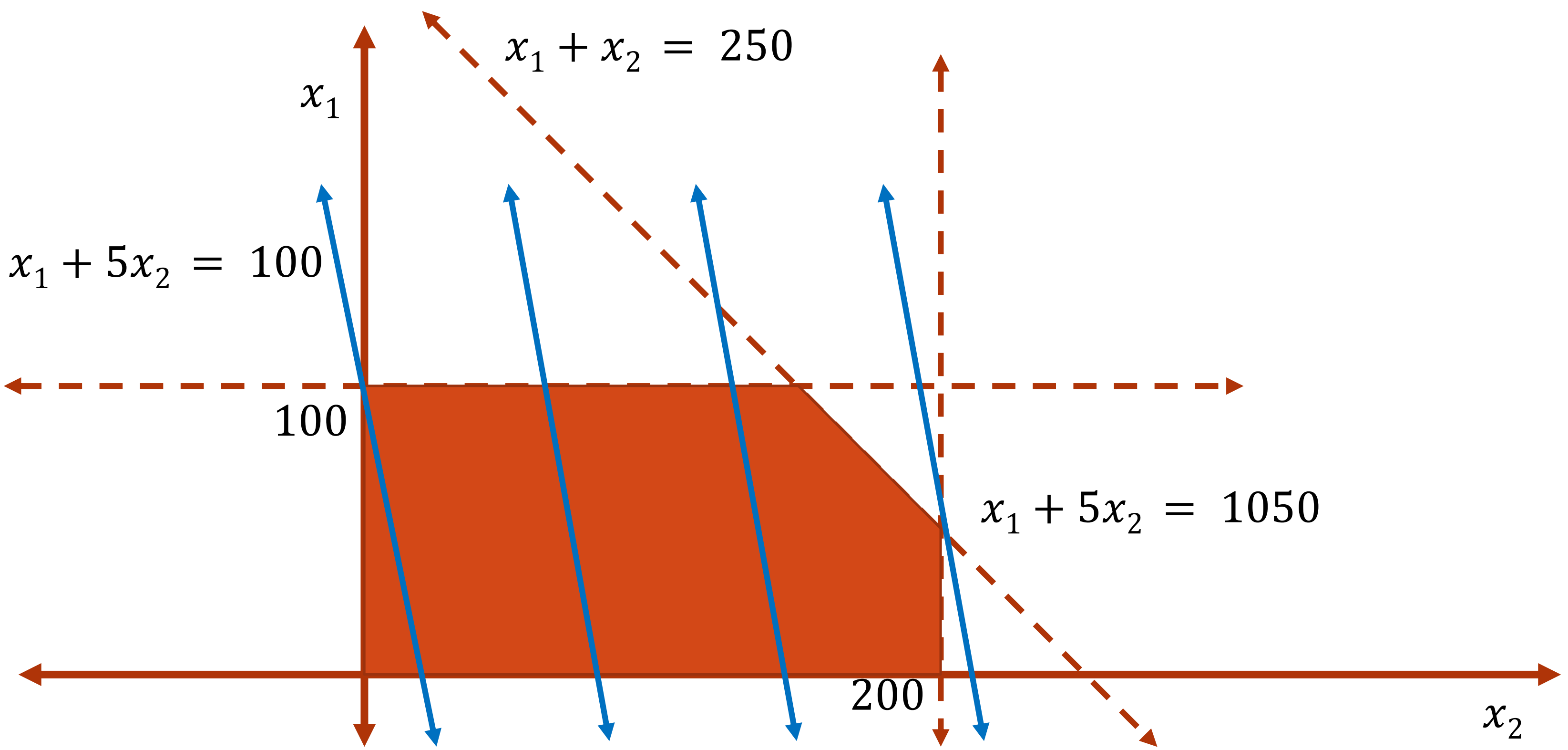
Linear Programming: Introduction

- Problem(LP): Maximize the *linear objective function:*

$$1 \cdot x_1 + 5 \cdot x_2$$

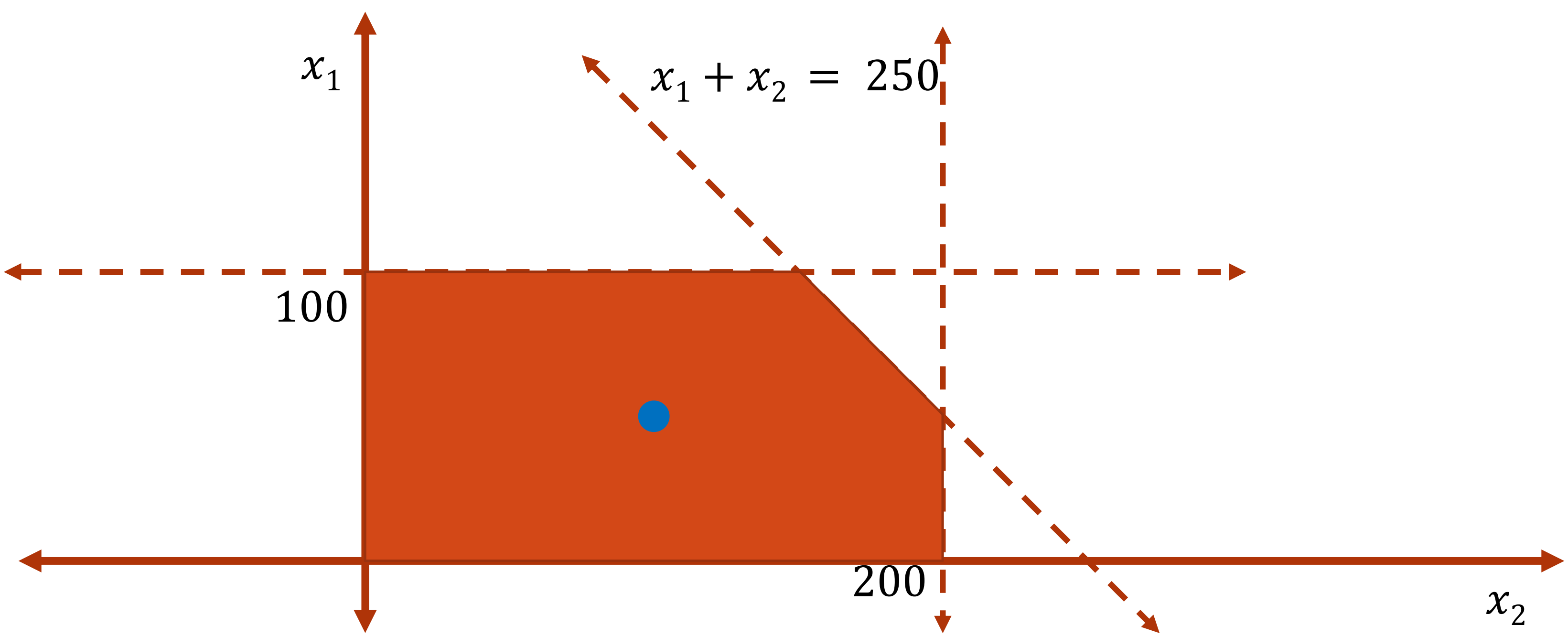
under the *linear constraints*:

$$x_1 \geq 0, x_2 \geq 0, x_1 \leq 100, x_2 \leq 200, x_1 + x_2 \leq 250$$



Linear Programming: Introduction

- Given a Linear Programming problem, we will use the following definitions:
  - Feasible solution: An assignment to the variables that satisfy all the linear constraints.
  - Example: $x_1 = 50, x_2 = 100$ is a feasible solution.



Linear Programming: Introduction

- Question: Does a Linear Programming problem always have a feasible solution?

Linear Programming: Introduction

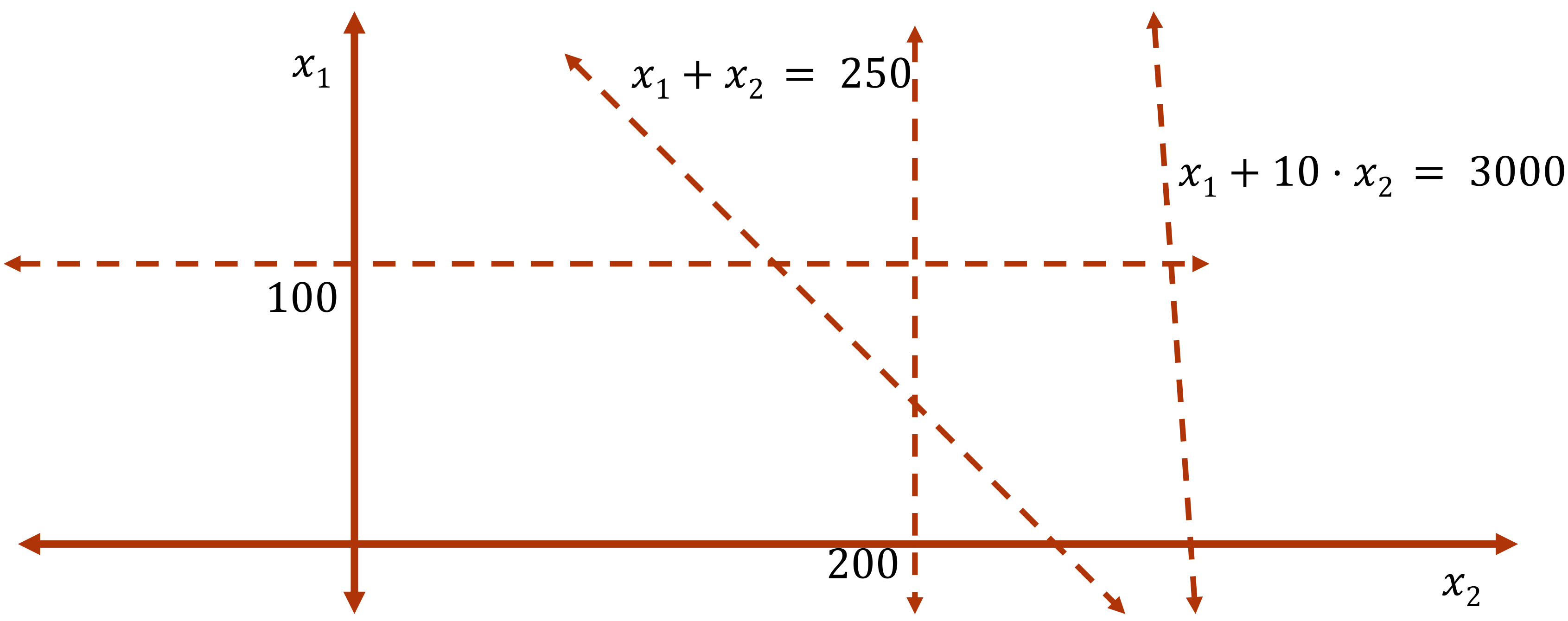
- Question: Does a Linear Programming problem always have a feasible solution?
  - Not necessarily. Suppose the linear constraints are

$$x_1 \geq 0, x_2 \geq 0, x_1 \leq 100, x_2 \leq 200,$$
$$x_1 + x_2 \leq 250, x_1 + 10 \cdot x_2 \geq 3000$$

Linear Programming: Introduction

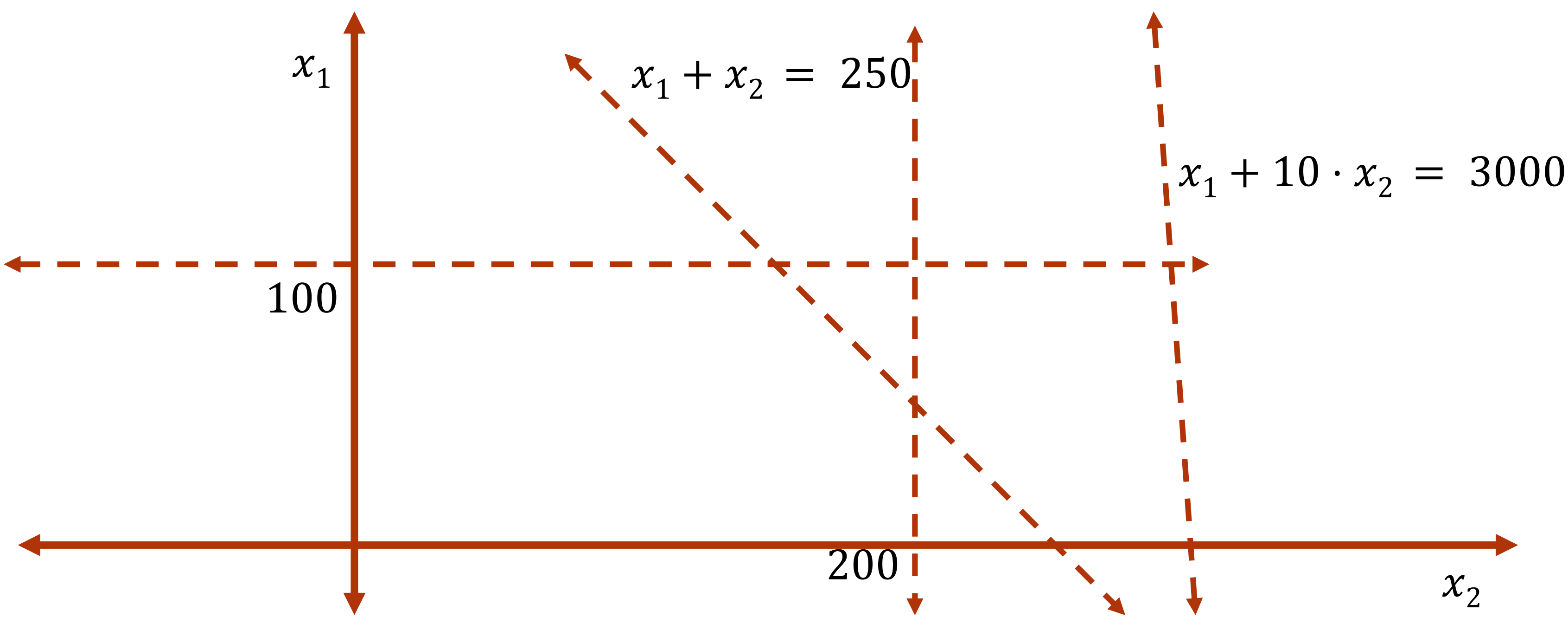
- Question: Does a Linear Programming problem always have a feasible solution?
  - Not necessarily. Suppose the linear constraints are

$$x_1 \geq 0, x_2 \geq 0, x_1 \leq 100, x_2 \leq 200,$$
$$x_1 + x_2 \leq 250, x_1 + 10 \cdot x_2 \geq 3000$$



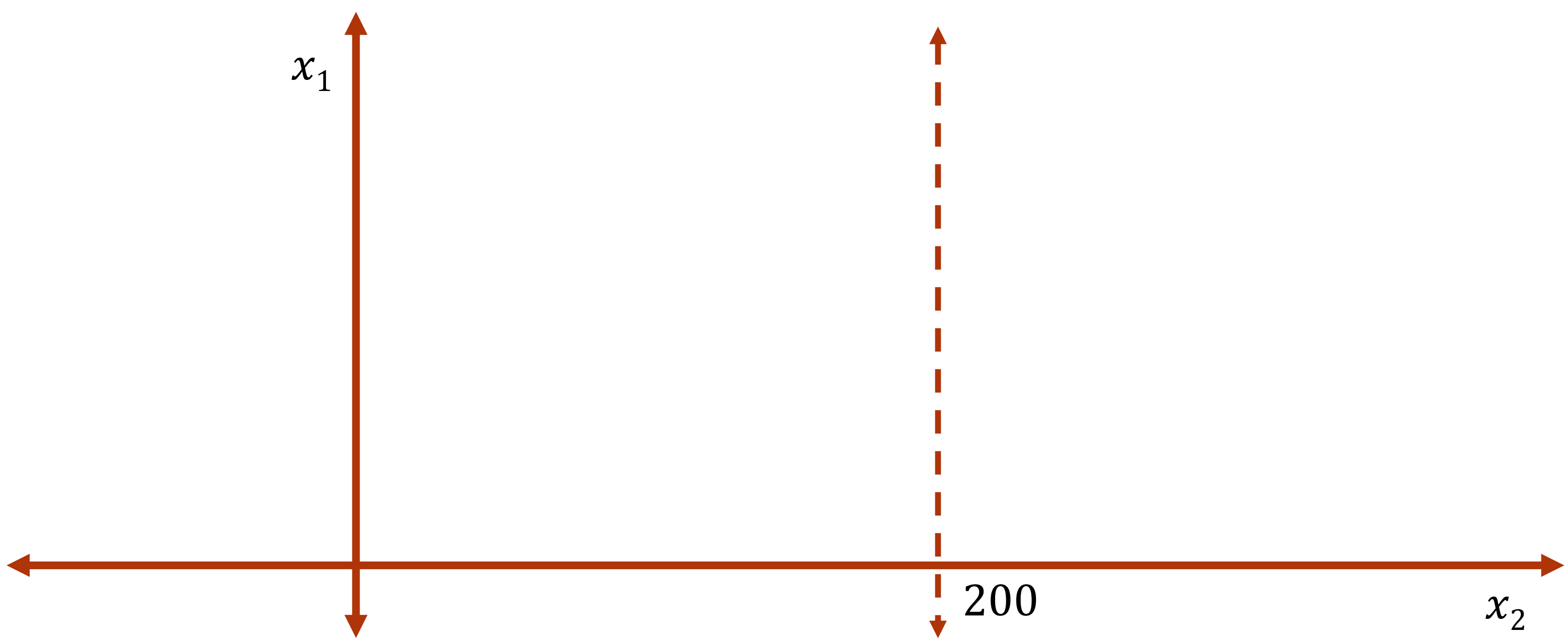
Linear Programming: Introduction

- <u>Infeasible LP</u>: A linear program is said to be infeasible if there are no feasible solutions.



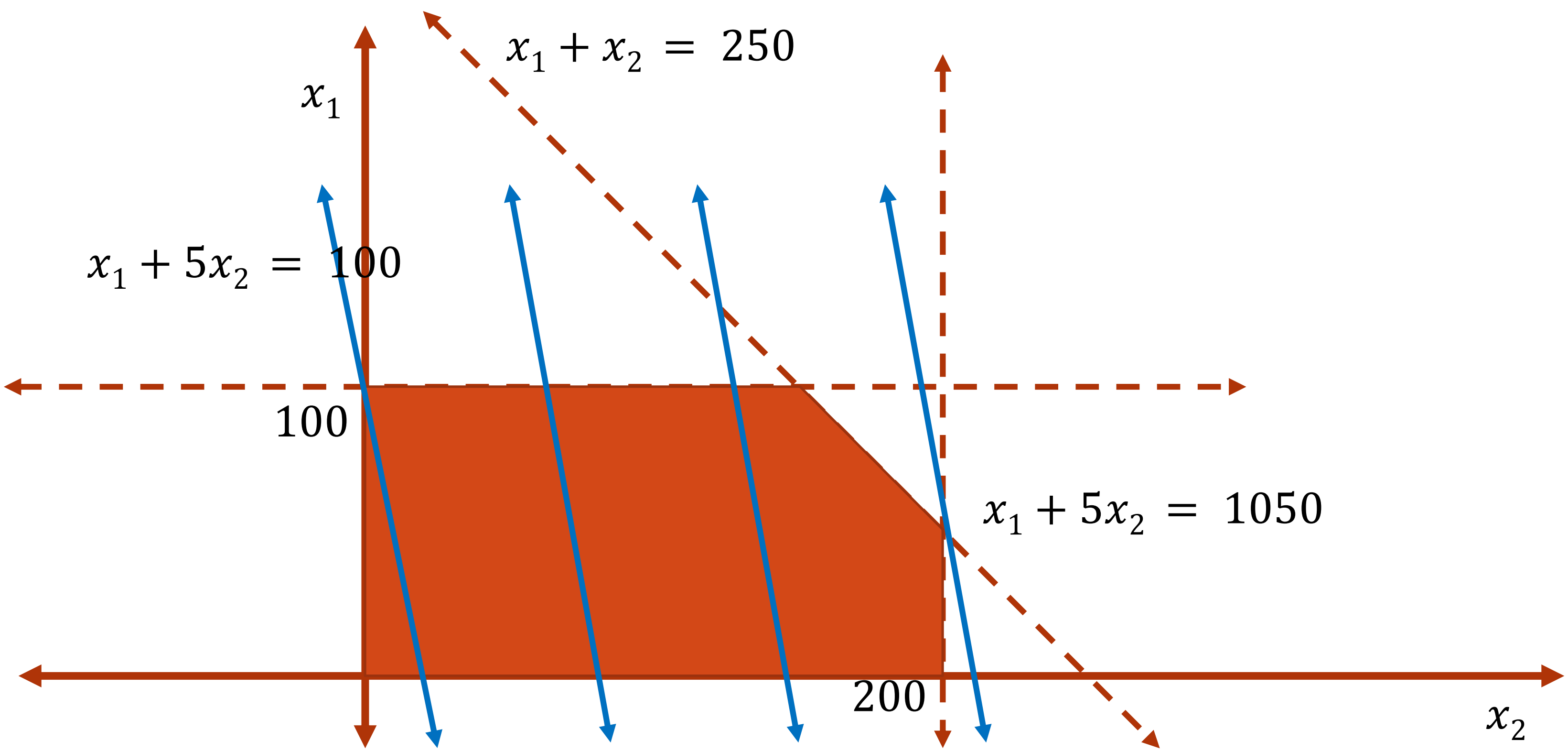
Linear Programming: Introduction

- Unbounded LP: A linear program is said to be unbounded if it is possible to achieve arbitrarily high values of the objective function.
  - Example: Maximize $(x_1 + 5 \cdot x_2)$
    subject to $x_1 \geq 0, x_2 \geq 0, x_2 \leq 200$.



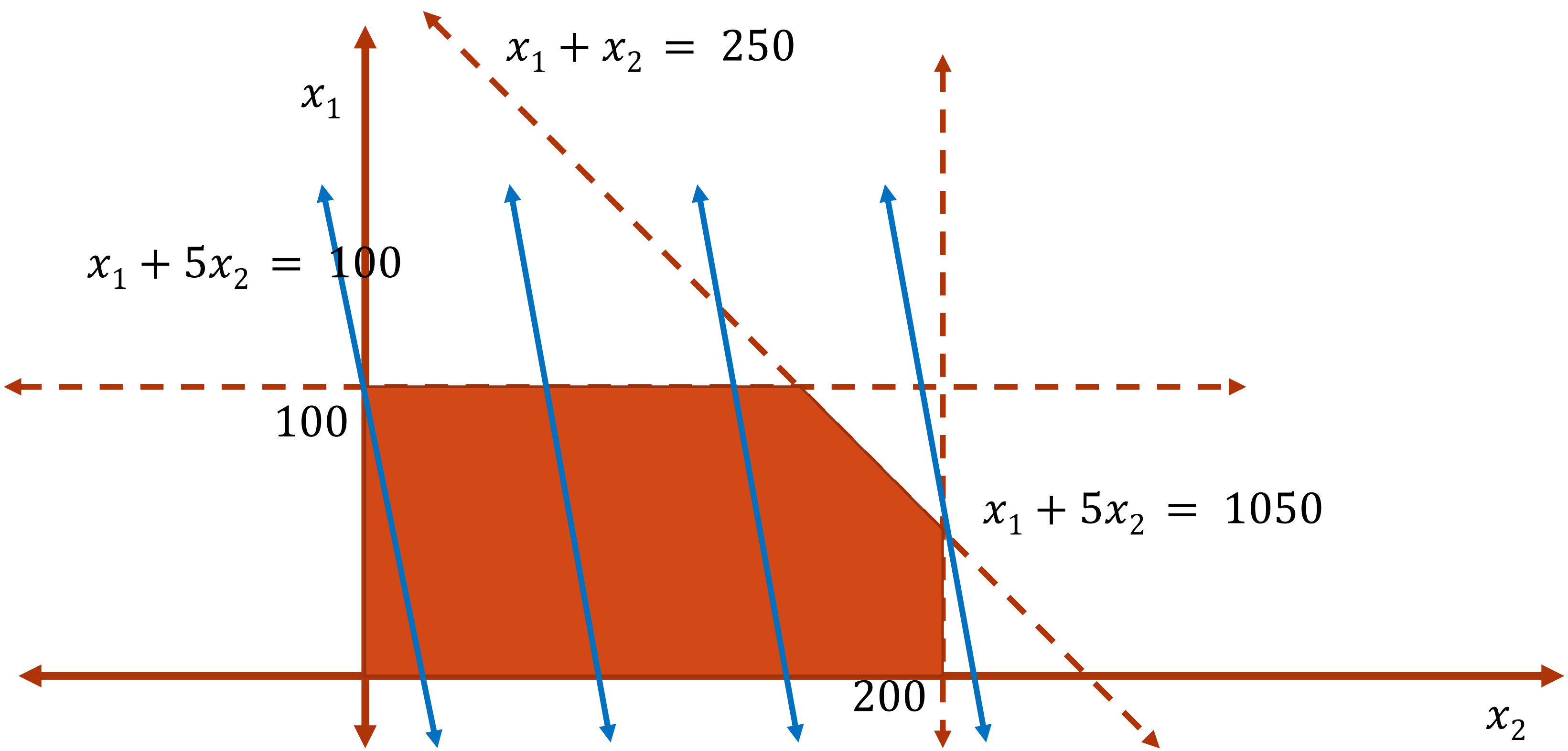
Linear Programming: Introduction

- <u>Claim</u>: For any linear program that is not infeasible and unbounded, the objective function value is maximized at one of the *vertices* of the feasible region.



Linear Programming: Introduction

- Naïve idea for solving an LP:
  - Try all possible vertex of the feasible region and return the one that maximizes the objective function.

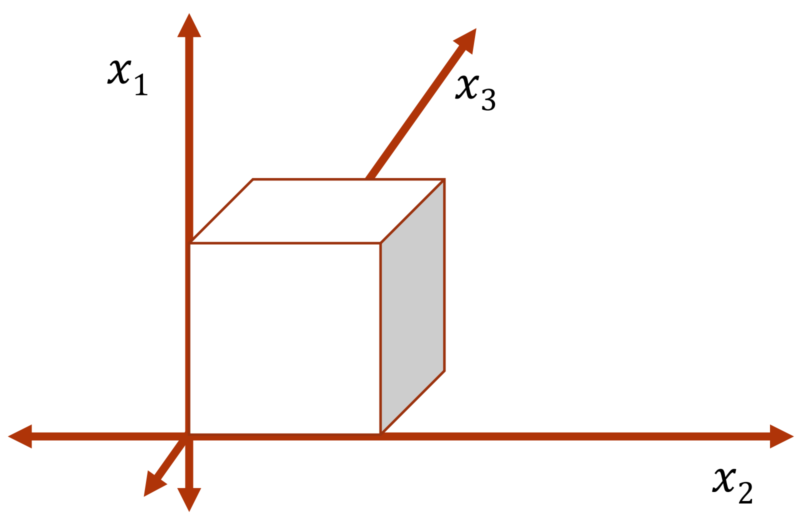


Linear Programming: Introduction

- Naïve idea for solving an LP:
  - Try all possible vertex of the feasible region and return the one that maximizes the objective function.
  - Suppose the LP has n variables and $m = O(n)$ constraints. How many vertices can the feasible region have in worst case?

Linear Programming: Introduction

- Naïve idea for solving an LP:
  - Try all possible vertex of the feasible region and return the one that maximizes the objective function.
  - Suppose the LP has n variables and $m = O(n)$ constraints. How many vertices can the feasible region have in worst case?
    - Exponentially many! Consider the LP: maximize $(x_1 + x_2 + \cdots + x_n)$ subject to $0 \le x_1, x_2, \dots, x_n \le 1$.

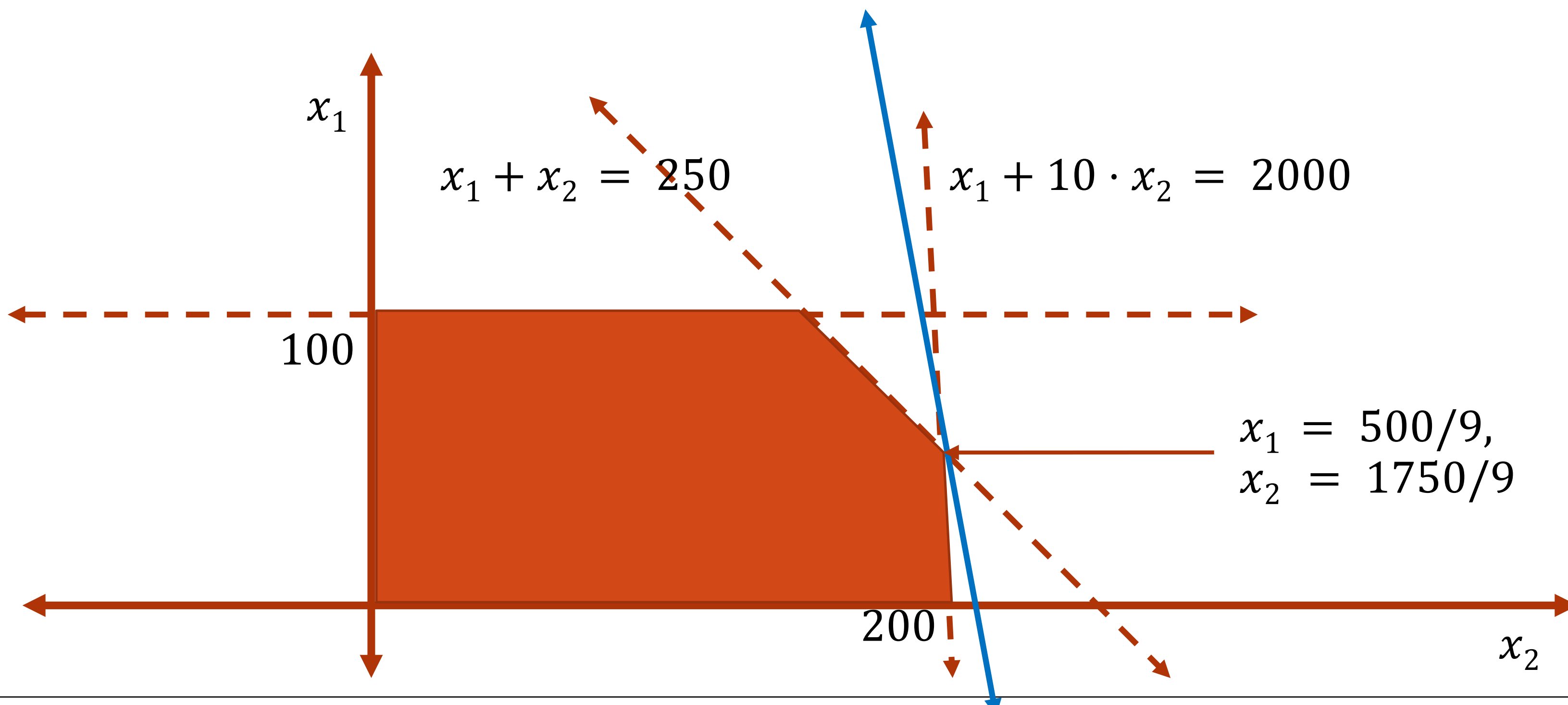


Linear Programming: Introduction

- Claim: There is an algorithm that solves any linear programming problem instance that runs in polynomial time.

Linear Programming: Introduction

- Claim: There is an algorithm that solves any linear programming problem instance that runs in polynomial time.
- The optimal solution may assign real numbers to some variables even though all of the constraints of objective function involve integers.



Linear Programming: Introduction

- Claim: There is an algorithm that solves any linear programming problem instance that runs in polynomial time.
- The optimal solution may assign real numbers to some variables even though all of the constraints of objective function involve integers.
- Suppose in addition to the linear constraint, we add another constraint that all the variables should be integers. Such linear programs are called Integer Linear Programs (ILP).
- Integer Linear Program(ILP): Consists of
  - Linear objective function
  - Linear constraints.
  - All variables should be integers.

    Decision-ILP: Given the above and an integer k, determine if there is an integer assignment to the variables such that the objective function value is at least k.

Linear Programming: Introduction

- How hard is Decision-ILP?

Linear Programming: Introduction

- How hard is Decision-ILP?
- Claim: Decision-ILP is **NP**-complete.
  - Proof:
    - Claim 1: Decision-ILP is in **NP**.
    - Claim 2: 3-SAT $\leq_p$ Decision-ILP

Linear Programming: Introduction

- How hard is Decision-ILP?
- Claim: Decision-ILP is **NP**-complete.
  - Proof:
    - Claim 1: Decision-ILP is in **NP**.
    - Claim 2: 3-SAT $\leq_p$ Decision-ILP
      - Proof idea: Given a 3-SAT formula, we construct an instance of Decision-ILP.
        For each clause (e.g., $(x_1 \vee x_2' \vee x_3)$) we create a linear constraint (e.g., $x_1 + 1 - x_2 + x_3 \geq 1$). We further consider constraints $0 \leq x_1, \ldots, x_n \leq 1$ and that all variables are integers.

Linear Programming: Introduction

- How hard is Decision-ILP?
- Claim: Decision-ILP is **NP**-complete.
  - Proof:
    - Claim 1: Decision-ILP is in **NP**.
    - Claim 2: 3-SAT $\leq_p$ Decision-ILP
      - Proof idea: Given a 3-SAT formula, we construct an instance of Decision-ILP.
        For each clause (e.g., $(x_1 \vee x_2' \vee x_3)$) we create a linear constraint (e.g., $x_1 + 1 - x_2 + x_3 \geq 1$). We further consider constraints $0 \leq x_1, \ldots, x_n \leq 1$ and that all variables are integers.
- Formulating problems as an ILP is a standard way of solving many combinatorial problems.
- Example: Maximum Independent set.
  - Consider a $0 - 1$ variable for each vertex, 1 denoting inclusion. For each edge (x, y), there is a constraint that $x + y \leq 1$.

Linear Programming

Solving problems by formulating as Linear Programs

Linear Programming: Applications

- We saw how some combinatorial problems can be formulated as an **Integer** Linear Programming (ILP) problem.
- Unfortunately, ILP is hard.
- A number of problems can be formulated as a Linear Programming problem and we know there is a polynomial time algorithm for LP.
- Some interesting applications:
  - Shortest $s - t$ path in a directed graph with non-negative weights.
  - Maximum flow in a network graph.

Linear Programming: Applications

- Problem (Maximum $s - t$ flow): Given a network graph $G = (V, E)$ with special source s and sink t, find the maximum value of an $s - t$ flow in the graph.
- Let $m = |E|$. We use m variables, one for each edge.
- For an edge (u, v), we will use variable f_{uv} to denote the flow along the edge (u, v).

Linear Programming: Applications

- Problem (Maximum $s-t$ flow): Given a network graph $G=(V,E)$ with special source s and sink t, find the maximum value of an $s-t$ flow in the graph.

- Let $m=|E|$. We use m variables, one for each edge.
- For an edge (u,v), we will use variable f_{uv} to denote the flow along the edge (u,v).
- We construct the following LP given G.
  - *Maximize* $\sum_{(s,v)\in E} f_{sv}$
  - Subject to,
    - $f_{uv} \leq c(u,v)$, for all (u,v) in E.
    - $\sum_{(v,u)\in E} f_{vu} = \sum_{(u,v)\in E} f_{uv}$, for all u in $V-\{s,t\}$.
    - $f_{uv} \geq 0$.

Linear Programming: Applications

- Problem (Shortest $s - t$ path): Given a weighted, directed graph $G = (V, E)$. Find the length of the shortest path from vertex s to vertex t.

Linear Programming: Applications

- Problem (Shortest $s - t$ path): Given a weighted, directed graph $G = (V, E)$. Find the length of the shortest path from vertex s to vertex t.

- Let $n = |V|$. We use n variables, one for each vertex.
- For a vertex v, we will use variable d_v to denote the length of the shortest path from vertex s to vertex v.

Linear Programming: Applications

- Problem (Shortest $s - t$ path): Given a weighted, directed graph $G = (V, E)$. Find the length of the shortest path from vertex s to vertex t.

- Let $n = |V|$. We use n variables, one for each vertex.
- For a vertex v, we will use variable d_v to denote the length of the shortest path from vertex s to vertex v.
- We construct the following LP given G.
  - *Maximize* d_t,
  - subject to:
    - For all edges $(u, v) \in E$, $d_v \leq d_u + w(u, v)$.
    - $d_s = 0$.

End