Mathematical Programming for All

Pano Santos, PhD

GUROBI OPTIMIZATION

The World's Fastest Solver

GUROBI OPTIMIZATION

Dedication

These video classes are dedicated to the memory of Saul I. Gass (1926-2013), a mentor and friend.

Goals

Goals

- These video classes are designed for an audience with limited or no background in mathematical programming, and includes: Data Scientists, Computer Scientists, Systems/IT Engineers, and Business Analysts.

  - **These video classes are designed for an audience with limited or no background in mathematical programming, and include**
  - **Data Scientists,**
  - **Computer Scientists,**
  - **Systems/IT Engineers,**
  - **Business Analysts.**

GUROBI OPTIMIZATION

Goals

- These video classes are designed for an audience with limited or no background in mathematical programming, and includes: Data Scientists, Computer Scientists, Systems/IT Engineers, and Business Analysts.

- The students taking this class will learn the foundations of Linear Programming (LP) and Mixed Integer Linear Programming (MIP).

  - **The students taking this class will learn the foundations of:**
  - **Linear Programming (LP)**
  - **Mixed Integer Linear Programming (MIP).**

GUROBI OPTIMIZATION

Goals

- These video classes are oriented to “students” with no or little background in mathematical programming, Data Scientists, Computer Scientists, Systems/IT Engineers, and Business Analysts.
- The students taking this class will learn the foundations of Linear Programming (LP) and Mixed Integer Linear Programming (MIP).
- You will learn how to formulate practical problems as Mixed Integer Linear Problems for: industrial, government, and military applications.

- **You will learn how to formulate practical problems as Mixed Integer Linear Problems for industrial, government, and military applications.**

GUROBI OPTIMIZATION

Goals

- These video classes are oriented to "students" with no or little background in mathematical programming, Data Scientists, Computer Scientists, Systems/IT Engineers, and Business Analysts.

Over 2,100 companies from approx. 50 industries use Gurobi for their mathematical programming applications

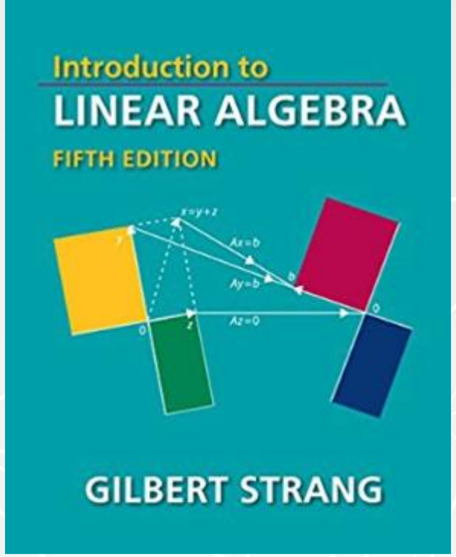
- The students taking this class will learn the foundations of Linear Programming (LP) and Mixed Integer Linear Programming (MIP).

- You will learn how to formulate practical problems as Mixed Integer Linear Problems for: industrial, government, and military applications.

Over 2,100 companies from approx. 50 industries use Gurobi for their mathematical programming applications

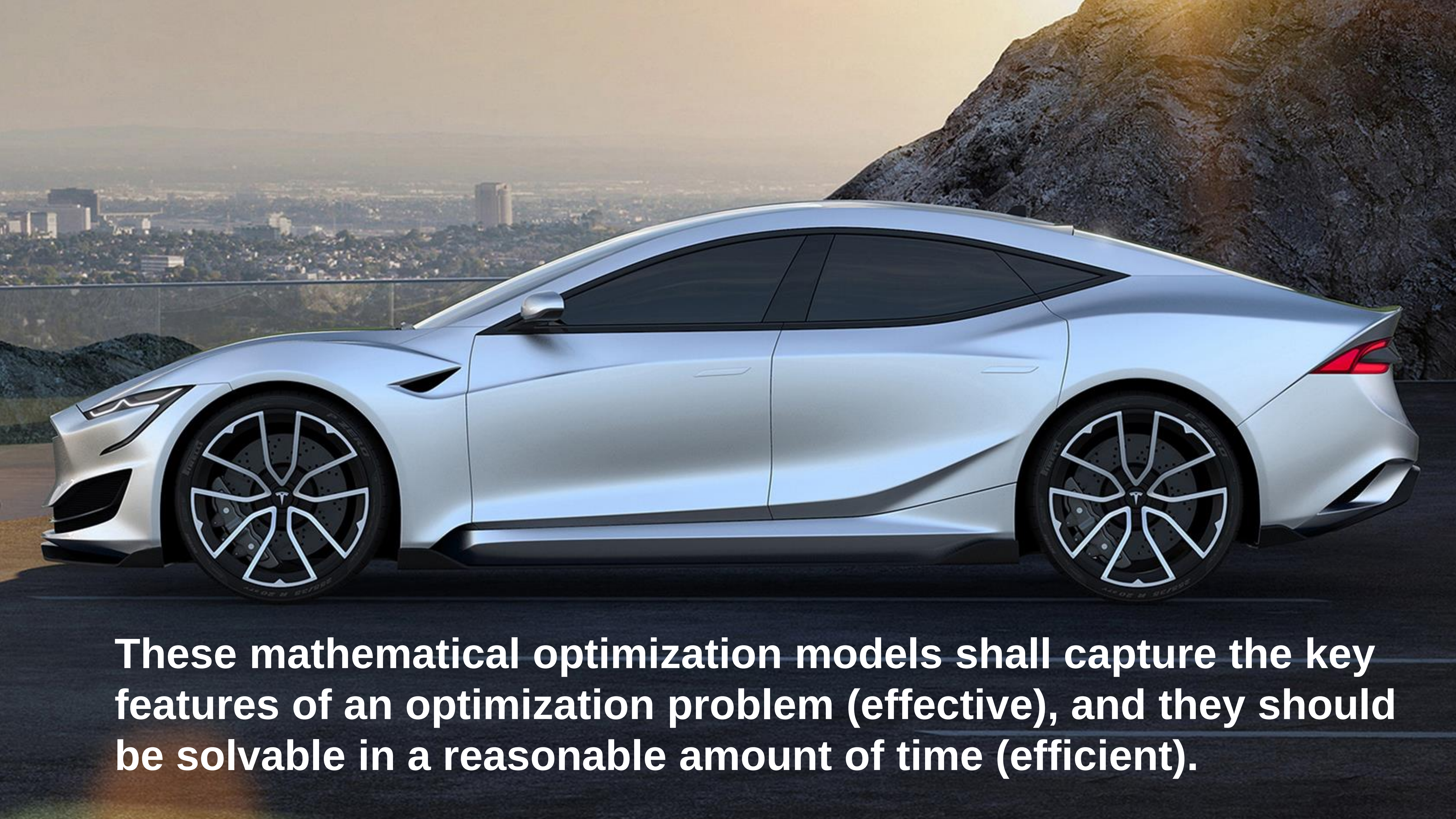
Prerequisites

Prerequisites



- **Linear algebra and calculus**
- **Both at the college level**
- **Familiarity with mathematical notation.**
  $e = mc^2$
- **Basic knowledge of Python**

Remarks


These mathematical optimization models shall capture the key features of an optimization problem (effective), and they should be solvable in a reasonable amount of time (efficient).

- **In spite of the pragmatic nature of these videos, it is important to cover theoretical aspects of Linear Programming and Integer Programming problems in order to build and tune up efficient mathematical optimization models.**

- **These series of introduction to mathematical programming videos will have three main chapters.**
    - Linear programming overview.
    - Mixed integer linear programming overview.
    - Mathematical programming model building overview.

- The duration of each video will be in the range of 10 min to 15 min.

Mathematical Programming

Background and relevance

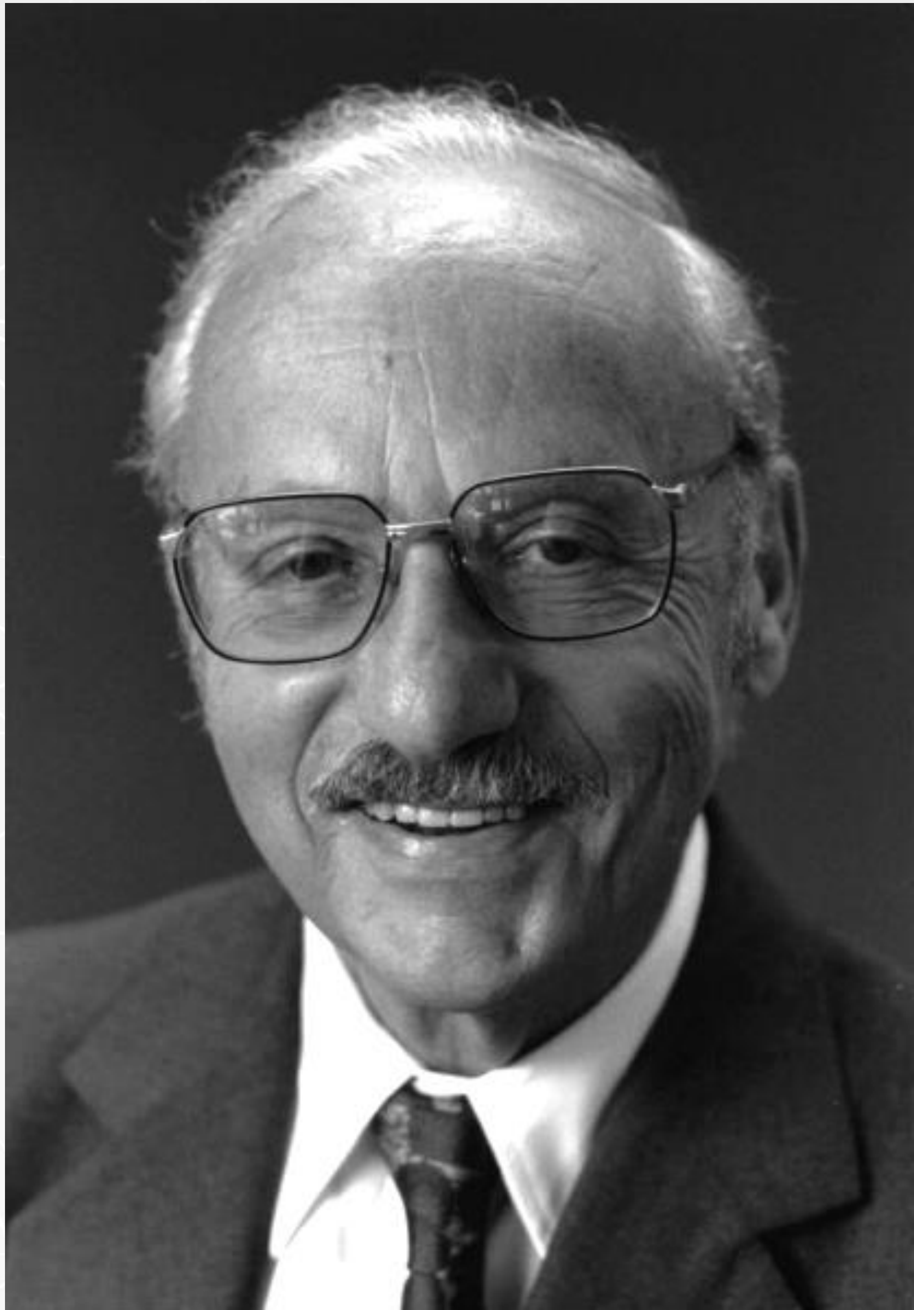
Origin of Mathematical Programming

Origin of Mathematical Programming

- **The origin of mathematical programming is the invention of linear programming in 1947, shortly after World War II.**

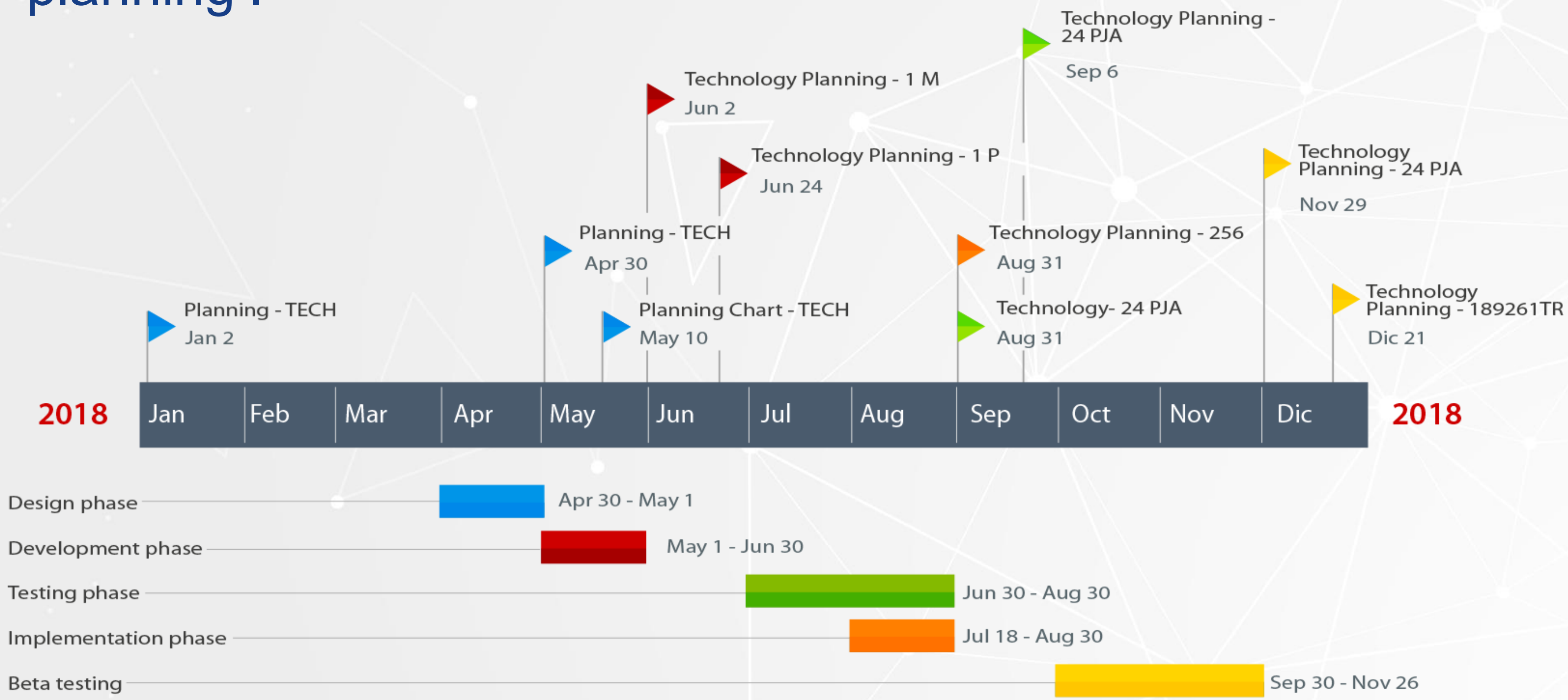
- **“Mathematical programming enables stating general goals and to lay out a path of detailed decisions to make in order to “best” achieve these goals when faced with a practical situation of great complexity”.** –George Dantzig

- **Mathematical programming entails**
  - the formulation of real-world problems in detailed mathematical terms (models).
  - the development of techniques for solving those models (algorithms).
  - and the use of SW and HW to develop applications.



Mathematical Programming Remarks

- **It should be pointed out that mathematical programming is different from computer programming.**
  - Mathematical programming is ‘programming’ in the sense of ‘planning’.

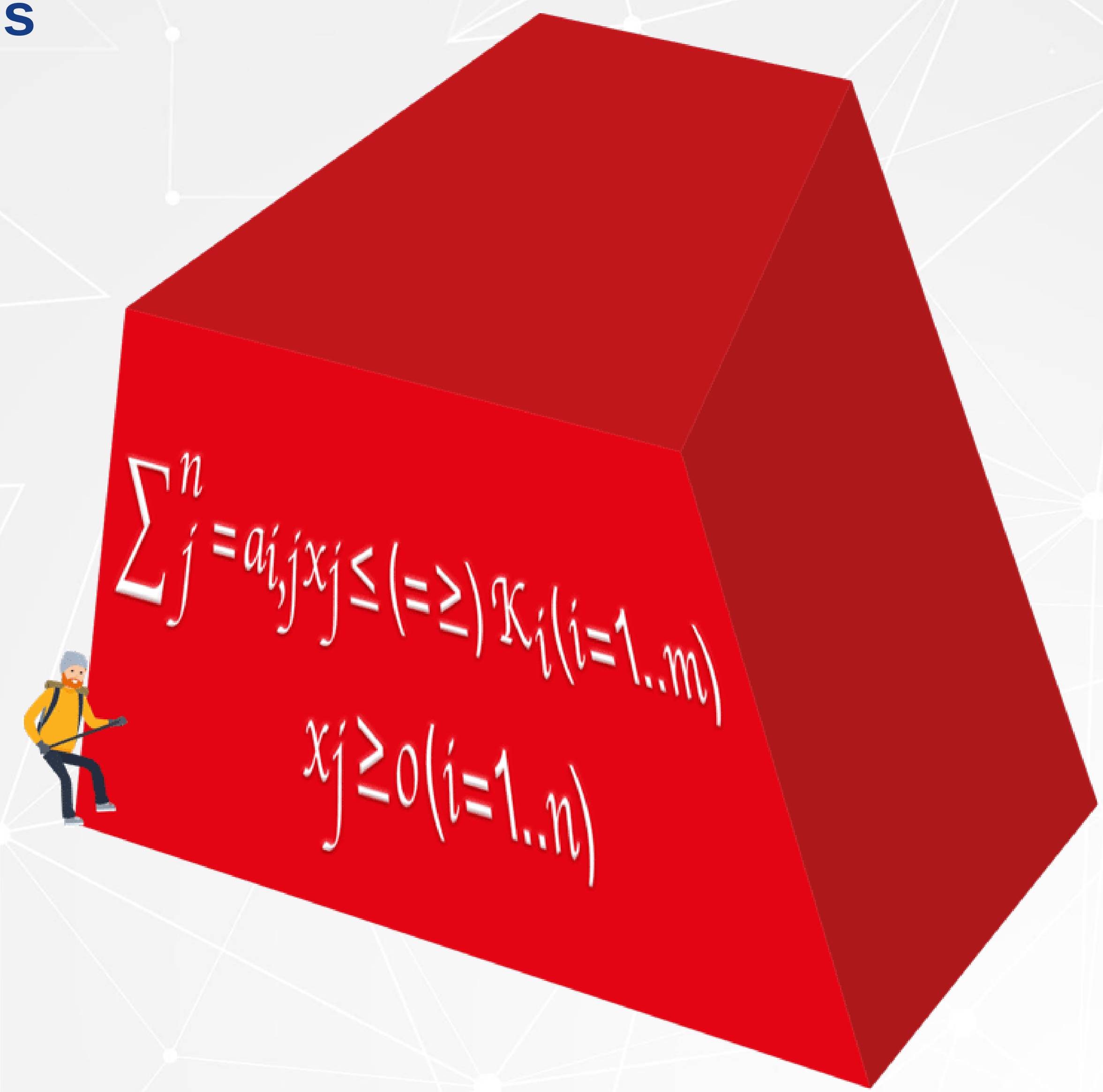


Mathematical Programming Remarks

- **The common feature that mathematical programming models have is that they all involve optimization.**
  - This is why mathematical programming is often called mathematical optimization.

Mathematical Programming Remarks

- **In these video classes, we focus on two special types of mathematical programming models.**
  - Linear Programming (LP) models.
  - Mixed Integer linear Programming (MIP) models.



Mathematical Programming Remarks

- **Mathematical programming is a declarative approach where the modeler formulates a mathematical optimization problem that captures the key features of a complex decision problem.**

- **Mathematical optimization formulations can then be solved by standard LP algorithms and MIP algorithms.**

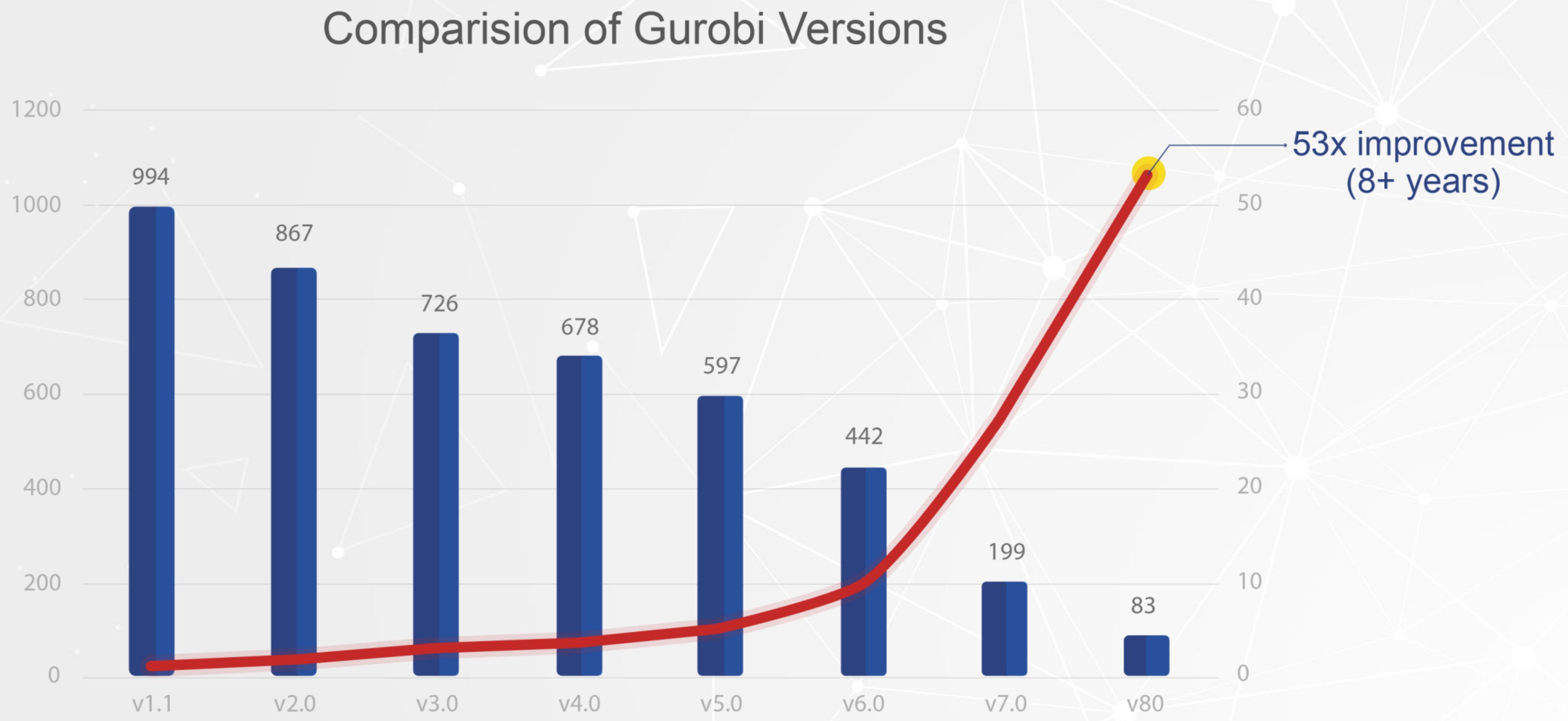
Mathematical Programming Remarks

- **Gurobi users formulate mathematical optimization problems that are solved by the Gurobi callable library.**
- **The mathematics and computer science behind Gurobi technology are leading edge.**
- **Gurobi has the best performance in the market.**

Mathematical Programming Remarks

- Gurobi users formulate mathematical optimization problems that are solved by the Gurobi callable library.

- The mathematics and computer science behind Gurobi technology are leading edge, that is why Gurobi solver has the best performance in the market.



GUROBI OPTIMIZATION

Mathematical Programming Remarks

- The particular implementation of the mathematics and computer science in the Gurobi Optimizer is quite complex.
- The user does not need to worry about how to solve the optimization problem at hand, this is done automatically by Gurobi behind the scenes.
- The user only needs to have an efficient LP or MIP model that captures the main characteristics of the optimization problem and the required data for the model.

GUROBI OPTIMIZATION

GUROBI
OPTIMIZATION

1. **Introduction to linear programming and mixed integer linear programming models - The furniture factory problem.**

    - Illustrative example prevalent throughout the video series.

    - Introduction to general formulations for linear programming and mixed integer programming problems.

2. Furniture factory problem - Graphical Solution

- How to graphically solve the furniture problem when formulated as a linear programming model.
- Introduction to important concepts related to the theory of linear programming.

3. Overview - Simplex method to solve linear programming problems.

- How the simplex method works.
- Key concepts of the theory of linear programming.

4. **Modeling and solving the furniture factory problem with the Gurobi python API.**

- How to use the Gurobi Python API to formulate the furniture problem as a linear programming problem and solve it using the Gurobi callable library

5. **Sensitivity analysis of LP problems with the Gurobi python API.**

- How linear programming models have an economic interpretation and the impact on the objective function value derived from marginal changes on a resource capacity value.

GUROBI OPTIMIZATION

6. Multiple optimal solutions, modeling opportunity with the Gurobi python API.

- How a linear programming problem can have multiple solutions.
- How having multiple solutions presents an opportunity to improve the linear programming problem formulation.

7. Unbounded solutions, modeling opportunity with the Gurobi python API.

- How a linear programming problem can be unbounded which means that the objective function value can be arbitrarily large.
- How an unbounded linear programming problem presents an opportunity to improve the linear programming problem formulation.

8. **Infeasible solutions, modeling opportunity with the Gurobi python API.**

- How a linear programming problem can be infeasible, lacking a solution that can satisfy all the constraints of the problem.
- How an infeasible linear programming problem presents an opportunity to improve the linear programming problem formulation.

9. **Maximize or minimize objective function.**

- How to tackle maximization and minimization linear programming problems.

10. **Unconstrained decision variables.**

- How Gurobi automatically handles unconstrained decision variables.

11. **Initial basic solution.**

- How to determine an initial solution of a linear programming problem in order to start the simplex method. This is done automatically by Gurobi.

GUROBI OPTIMIZATION

12. Presolve.

- An example of how Presolve reduces the size of a linear programming problem. Gurobi by default calls Presolve to significantly reduce the size of an LP problem.

13. Matrix sparsity.

- An important characteristic of linear programming problems that is related to the number of non-zero coefficients associated with the variables in the problem formulation.

GUROBI
OPTIMIZATION

14. Duality in linear programming.

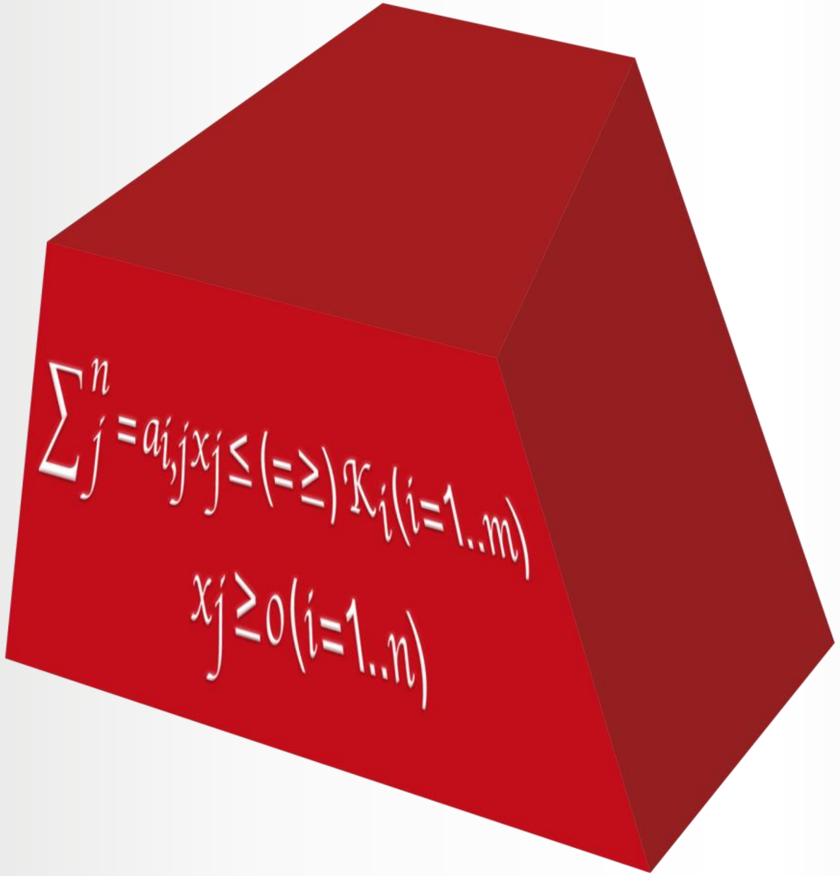
- One of the most important concepts in linear programming that allows the efficient characterization of optimal solutions of a linear programming problem.

15. Optimality conditions in linear programming.

- Discussion of duality and establish sufficient and necessary conditions of optimal solutions of a linear programming problem.

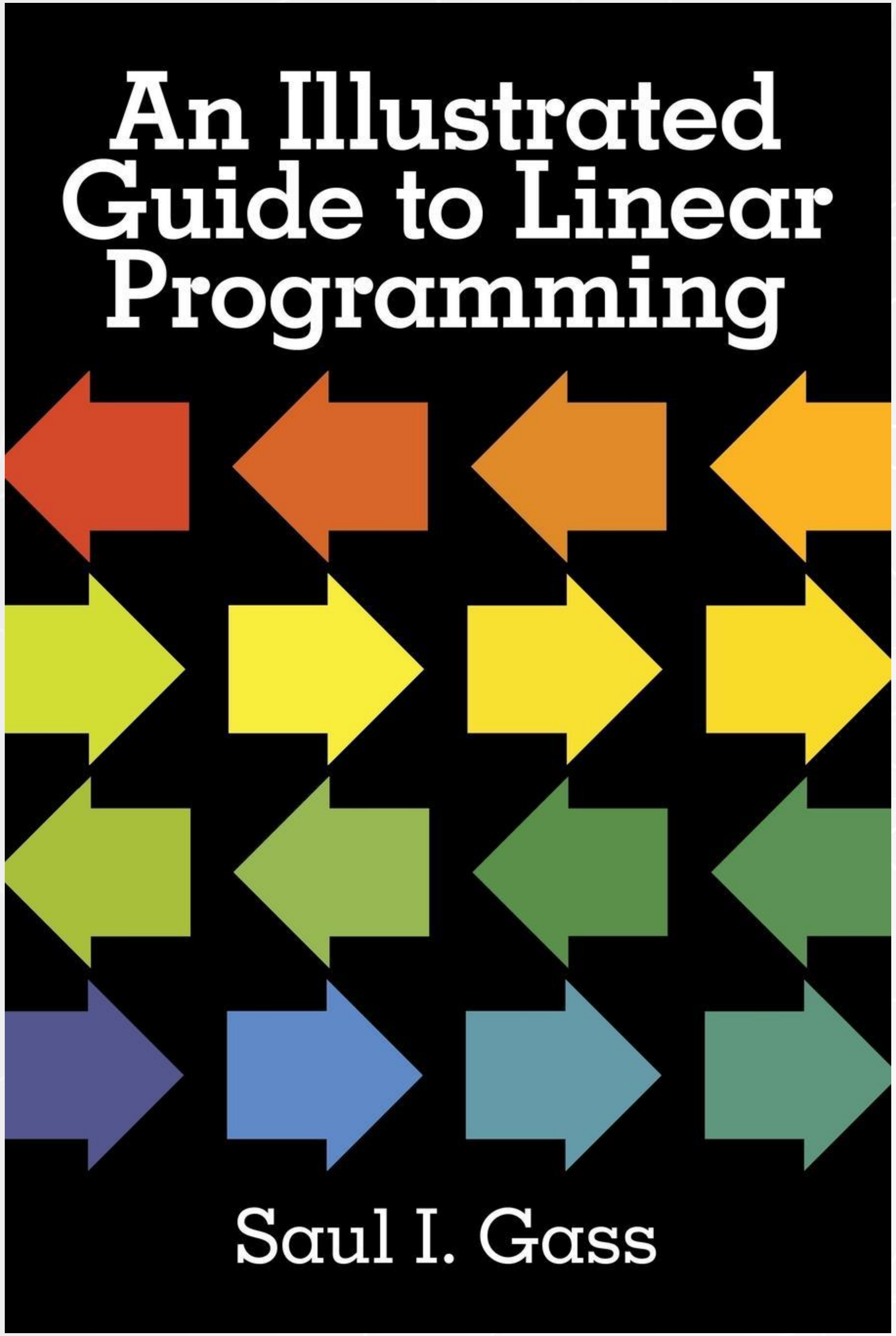
16. Dual simplex method.

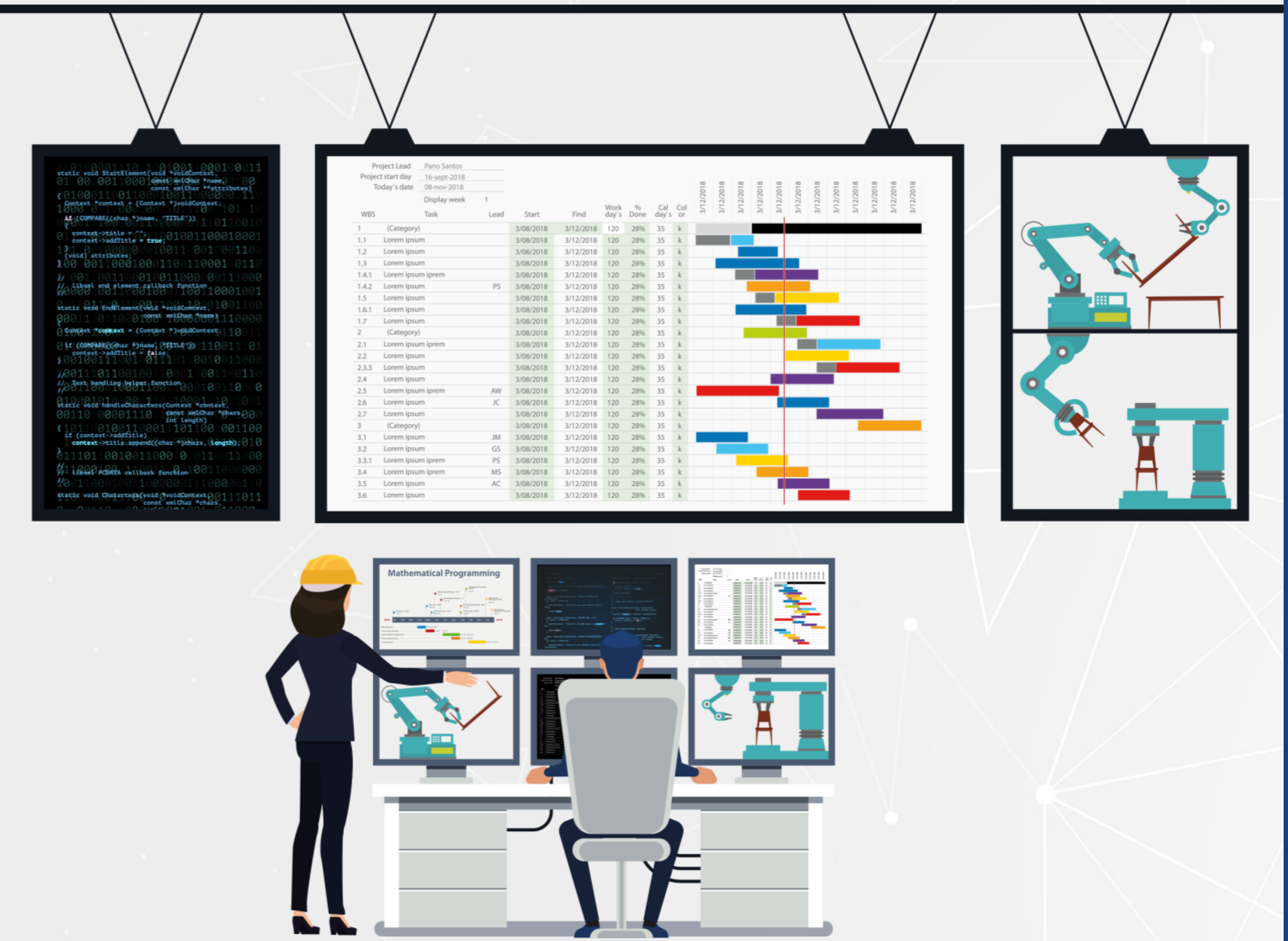
- A variation of the simplex method that is frequently used to solve mixed integer linear programming problems.



Introduction to linear programming

The Furniture Factory problem


An Illustrated
Guide to Linear
Programming
Saul I. Gass



The Furniture Factory Problem

A data scientist is in charge of developing the Weekly Production Plan of two key products that the furniture factory makes: chairs and tables.

The data scientist using machine learning techniques predicts that the selling price of a chair is $45 and the selling price of a table is $80 dollars.

GUROBI OPTIMIZATION

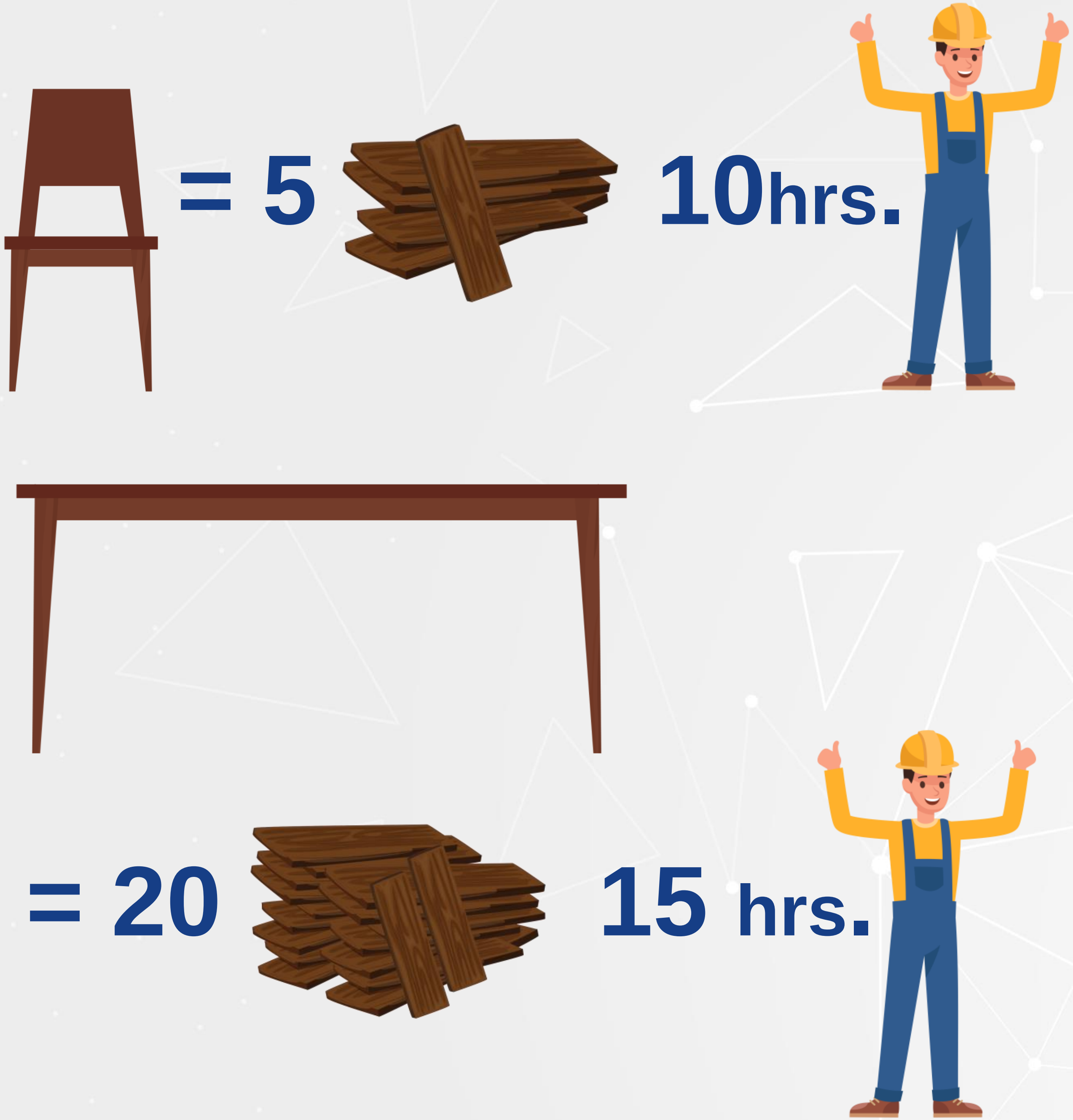
The Furniture Factory Problem

A data scientist is in charge of developing the weekly production plan of two key products that the furniture factory makes: chairs and tables.

The data scientist using machine learning techniques predicts that the selling price of a chair is $45 and the selling price of a table is $80 dollars.

- **There are two critical resources in the production of chairs and tables:**
  - Mahogany (measured in board square-feet) and labor (measured in work hours).
  - There are 400 units of mahogany available at the beginning of each week.
  - There are 450 units of labor available during each week.

GUROBI
OPTIMIZATION



The Furniture Factory Problem

A data scientist is in charge of developing the weekly production plan of two key products that the furniture factory makes: chairs and tables.
The data scientist using machine learning techniques predicts that the selling price of a chair is $45 and the selling price of a table is $80 dollars.

- **There are two critical resources in the production of chairs and tables:**
  - Mahogany (measured in board square-feet) and labor (measured in work hours).
  - There are 400 units of mahogany available at the beginning of each week.
  - There are 450 units of labor available during each week.

- **The data scientist estimates that**
  - One chair requires 5 units of mahogany and 10 units of labor.
  - One table requires 20 units of mahogany and 15 units of labor.
- **The marketing department has told the data scientist that <u>ALL</u> the production of chairs and tables can be sold.**

GUROBI OPTIMIZATION

Problem statement:
What is the Production Plan that maximizes total revenue?

The Furniture Factory Problem

A data scientist is in charge of developing the weekly production plan of two key products that the furniture factory makes: chairs and tables.
The data scientist using machine learning techniques predicts that the selling price of a chair is $45 and the selling price of a table is $80 dollars.

- **There are two critical resources in the production of chairs and tables:**
  - Mahogany (measured in board square-feet) and labor (measured in work hours).
  - There are 400 units of mahogany available at the beginning of each week.
  - There are 450 units of labor available during each week.

- **The data scientist estimates that**
  - One chair requires 5 units of mahogany and 10 units of labor.
  - One table requires 20 units of mahogany and 15 units of labor.
- **The marketing department has told the data scientist that all the production of chairs and tables can be sold.**

GUROBI OPTIMIZATION

The Furniture Factory Problem… 2

DATA	CHAIR	TABLE	CAPACITY
	5 units	20 units	400 units
	10 hours	15 hours	450 hours
PRICE \$	\$45	\$80	

Note: for the moment, we will assume we can produce and sell fractional quantities of a chair or table. In chapter 2 of these videos series we show how to tackle mathematical programming problems where you require that the decision variables most be integer numbers.

The data of the furniture problem can be summarized in the following table:

- **To determine a Production Plan, we need to decide how many chairs and tables to make in order to maximize total revenue, while satisfying resources constraints.**
- **This problem has two decision variables:**
  - x1: number of chairs to produce.
  - x2: number of tables to produce.
  - The number of chairs and tables to produce should be a non-negative number. That is, x1, x2 $\geq$ 0.

The Furniture Factory Problem… 2

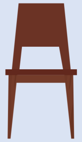
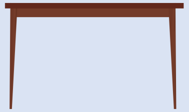
- If we would know the value of the amount of chairs to produce (x1), then since each chair generates $45, the total revenue generated by the production of chairs can be determined by the term 45x1 (45*x1).

- Similarly, the total revenue generated by the production of tables can be determined by the term 80x2 (80*x2).

- Consequently, the total revenue generated by the production plan can be determined by the following equation.

  - **Revenue = 45x1 + 80x2**

The data of the furniture problem can be summarized in the following table:

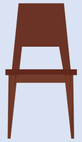
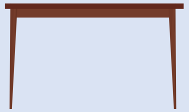
 DATA	 CHAIR	 TABLE	 CAPACITY
	5 units	20 units	400 units
	10 hours	15 hours	450 hours
	$45	$80	

The Furniture Factory Problem… 2

- **The Production Plan is constrained by the amount of resources available.**
- **How do we ensure that the production plan does not consume more mahogany than the amount of mahogany available?**
- **If we decide to produce x1 number of chairs, then the total amount of mahogany consumed by the production of chairs is 5x1 (5*x1).**
- **Similarly, if we decide to produce x2 number of tables, then the total amount of mahogany consumed by the production of tables is 20x2 (20*x2).**
- **Hence, the total consumption of mahogany by the production plan determined by the values of x1 and x2 is (5x1 + 20x2). However, the consumption of mahogany by the production plan cannot exceed the amount of mahogany available. We can expressed these ideas in the following constraint:**

$$5x1 + 20x2 \leq 400$$

The data of the furniture problem can be summarized in the following table:

 DATA	 CHAIR	 TABLE	 CAPACITY
	5 units	20 units	400 units
	10 hours	15 hours	450 hours
	$45	$80	

GUROBI OPTIMIZATION

- The production plan is constrained by the amount of resources available.
- In similar fashion, we can formulate the constraint for labor resources.
- The total amount of labor resources consumed by the production of chairs is 10 labor units multiplied by the number of chairs produced, that is 10x1 (10*x1).
- The total amount of labor resources consumed by the production of tables is 15 labor units multiplied by the number of tables produced, that is 15x2 (15*x2).
- Therefore, the total consumption of labor resources by the production plan determined by the values of x1 and x2 is (10x1 + 15x2). This labor consumption cannot exceed the labor capacity available. Hence, this constraint can be expressed as follows:

$$10x1 + 15x2 \leq 450.$$

The Furniture Factory Problem… 2

The data of the furniture problem can be summarized in the following table:

DATA	CHAIR	TABLE	CAPACITY
	5 units	20 units	400 units
	10 hours	15 hours	450 hours
PRICE $	$45	$80	

- **The furniture problem formulation as a linear programming (LP) problem is**

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Units of mahogany capacity}$$

$$(3.0).\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor hours capacity}$$

$$\boldsymbol{x_1, x_2 \geq 0} \quad \text{Non} - \text{negativity}$$

The Furniture Factory Problem… 2

The data of the furniture problem is

DATA	CHAIR	TABLE	CAPACITY
	5 units	20 units	400 units
	10 hours	15 hours	450 hours
	$45	$80	

Introduction linear programming and mixed integer linear programming problems

Key components of mathematical programming models

x_1 is the decision variable representing the number of chairs to produce. The index 1 refers to the product chair.

x_2 is the decision variable representing the number of tables to produce. The index 2 refers to the product table.

We can create a set of products mapping each product with the index associated with each decision variable. Then, the set products = {1: chair, 2: table} maps each index with its corresponding product.

Similarly, we can create a set for resources as follows: resources = {1: mahogany, 2: labor} where the index 1 maps to the resource mahogany, and the index 2 maps to the resource labor.

Key components of linear programming model

$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$

$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Units of mahogany capacity}$

$(3.0).\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor hours capacity}$

$\boldsymbol{x_1, x_2 \geq 0} \quad \text{Non} - \text{negativity}$

Key components of linear programming model… 2

This LP model has several types of parameters representing known quantities (data) that characterize the problem.

- Prices can be defined over the set of products, e.g. b_1 = 45 means that the price of a chair is $45, and b_2 = 80 means that the price of a table is $80.

- Resources capacity can be defined over the set of resources, e.g. K_1 = 400, means that the availability of mahogany is 400 units/week, and K_2 = 450 means that the availability of labor is 450 units/week.

- Technology coefficients describe the consumption of resources when building a product. For example,

  $\mathrm{a}_{1,1}$ = 5 means that five units of mahogany are consumed when building one chair,

  $\mathrm{a}_{1,2}$ = 20 means that twenty units of mahogany are consumed when building one table,

  $\mathrm{a}_{2,1}$ = 10 means that ten units of labor are consumed when building one chair,

  $\mathrm{a}_{2,2}$ = 15 means that fifteen units of labor are consumed when building one table.

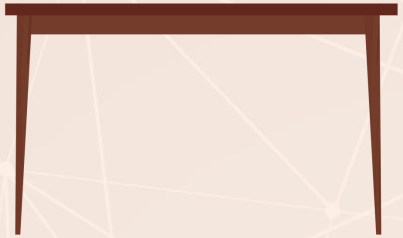
$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Units of mahogany capacity}$$

$$(3.0).\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor hours capacity}$$

$$\boldsymbol{x_1, x_2 \geq 0} \quad \text{Non} - \text{negativity}$$

- **Matrix of technology coefficients**

 DATA	 CHAIR	 TABLE
 Mahogany	$a11 = 5$	$a12 = 20$
 Labor	$a21 = 10$	$a22 = 15$

GUROBI OPTIMIZATION

Key components of linear programming model… 3

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Units of mahogany capacity}$$

$$(3.0).\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor hours capacity}$$

$$\boldsymbol{x_1, x_2 \geq 0} \quad \text{Non} - \text{negativity}$$

This LP model has two types of **constraints** limiting the number of chairs and tables that can be produced.

These constraints are defined over the set of resources, and represent that the consumption of each resource by a Production Plan cannot exceed the amount available of the resource.

The **objective function** is to maximize total revenue generated by the optimal Production Plan.

Linear Programming model components

Summary

SET OF INDICES

- Set of resources = {1: mahogany, 2: labor}
- Set of products = {1: chairs, 2: tables}

GUROBI OPTIMIZATION

Abstraction and generalization of furniture problem

Original LP problem formulation

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \le 400 \quad \text{Units of mahogany capacity}$$

$$(3.0).\quad 10x_1 + 15x_2 \le 450 \quad \text{Labor hours capacity}$$

$$x_1, x_2 \ge 0 \quad \text{Non} - \text{negativity}$$

Parametrized LP problem formulation

$$(1.0).\quad \text{Max } (b_1 = 45)x_1 + (b_2 = 80)x_2$$

$$(2.0)\quad (a_{1,1} = 5)x_1 + (a_{1,2} = 20)x_2 \le K_1 = 400$$

$$(3.0).\quad (a_{2,1} = 10)x_1 + (a_{2,2} = 15)x_2 \le K_2 = 450$$

$$x_1, x_2 \ge 0$$

Parametrization of input data

Parametrization of input data of an LP problem allows one to separate the data from the model. That is, one can change the values of the data without changing the model.

Linear Programming model components

Summary

SET OF INDICES
- Set of resources = {1: mahogany, 2: labor}
- Set of products = {1: chairs, 2: tables}

PARAMETERS
- Product prices.
- Resources capacity.
- Technology coefficients.

DECISION VARIABLES
- x1: number of chairs to make,
- x2: number of tables to make

CONSTRAINTS
- Amount of mahogany available per week.
- Amount of labor available per week.

OBJECTIVE FUNCTION
- Maximize total weekly revenue.

Abstraction and generalization of furniture problem… 2

Parametrized LP problem formulation

Prices

$$(1.0) \qquad Max \quad b_1x_1 + b_2x_2$$

$$(2.0) \qquad a_{1,1}x_1 + a_{1,2}x_2 \leq K_1$$

$$(3.0) \qquad a_{2,1}x_1 + a_{2,2}x_2 \leq K_2$$

$$\boldsymbol{x_1, x_2 \geq 0}$$

Resources capacity

Technology coefficients

$$b_1x_1 + b_2x_2 = \sum_{j=1}^{2} b_jx_j$$

$$\sum_{j=1}^{2} a_{i,j}x_j \leq K_i \ \ (i = 1,2)$$

$$x_j \geq 0, (j = 1,2)$$

Abstraction and generalization of furniture problem… 2

General LP problem formulation

Objective function coefficients

Objective Function

$$Max \sum_{j=1}^{n} b_j x_j$$

Decision Variables

RHS: right hand side

Constraints

$$\sum_{j=1}^{n} a_{i,j} x_j \leq K_i \ (i = 1..m)$$

Set of indices for Constraints

Technology coefficients

$$x_j \geq 0 \ \ (j = 1..n)$$

Set of indices for Variables

n = (2) number of decision variables (products)
m = (2) number of constraints (resources)

Parametrized LP problem formulation

$$(1.0) \quad Max \quad b_1x_1 + b_2x_2$$

$$(2.0) \quad a_{1,1}x_1 + a_{1,2}x_2 \leq K_1$$

$$(3.0) \quad a_{2,1}x_1 + a_{2,2}x_2 \leq K_2$$

$$\boldsymbol{x_1, x_2 \geq 0}$$

$$b_1x_1 + b_2x_2 = \sum_{j=1}^{2} b_j x_j$$

$$\sum_{j=1}^{2} a_{i,j} x_j \leq K_i \ \ (i = 1,2)$$

$$x_j \geq 0, (j = 1,2)$$

Linear, integer, binary, mixed programming models

Linear programming problems

Objective function examples:

- maximize total revenue
- minimize total cost.

$$Max\ (Min) \sum_{j=1}^{n} b_j xj$$

Constraint examples:

- $\leq$ constraints are typically considered for capacity constraints where you don't want to exceed capacity available.
- $\geq$ constraints are used to model demand requirements where you want to ensure that at least certain level of demand is satisfied.
- $=$ constraints are used when you want to match exactly certain activities with a given requirement. For example, a job position can only be filled with one resource, and you have a set of possible qualified resources to assign to the job.

$$\sum_{j=1}^{n} a_{i,j} x_j \leq (=, \geq)\ K_i\ (i = 1 \ldots m)$$

$$x_j \geq 0\ \ (j = 1..n)$$

Linear, integer, binary, mixed programming models

Linear programming problems
Integer programming problems

Objective function examples:

- maximize total revenue
- minimize total cost.

Constraint examples:

- $\leq$ constraints are typically considered for capacity constraints where you don't want to exceed capacity available.
- $\geq$ constraints are used to model demand requirements where you want to ensure that at least certain level of demand is satisfied.
- $=$ constraints are used when you want to match exactly certain activities with a given requirement. For example, a job position can only be filled with one resource, and you have a set of possible qualified resources to assign to the job.

$$Max\ (Min) \sum_{j=1}^{n} b_j xj$$

$$\sum_{j=1}^{n} a_{i,j} x_j \leq (=, \geq)\ K_i\ (i = 1..m)$$

$$x_j \geq 0\ \ and\ integer\ \ (j = 1..n)$$

GUROBI OPTIMIZATION

Linear, integer, binary, mixed programming models

Binary programming problems
Integer programming problems

Objective function examples:

- maximize total revenue
- minimize total cost.

$$Max\ (Min) \sum_{j=1}^{n} b_j xj$$

Constraint examples:

- $\leq$ constraints are typically considered for capacity constraints where you don't want to exceed capacity available.
- $\geq$ constraints are used to model demand requirements where you want to ensure that at least certain level of demand is satisfied.
- $=$ constraints are used when you want to match exactly certain activities with a given requirement. For example, a job position can only be filled with one resource, and you have a set of possible qualified resources to assign to the job.

$$\sum_{j=1}^{n} a_{i,j} x_j \leq (=, \geq)\ K_i \ (i = 1 \ldots m)$$

$$x_j \ \ in \ \{0,1\} \ \ (j = 1..n)$$

Linear, integer, binary, mixed programming models

Mixed integer linear programming problems

Objective function examples:

- maximize total revenue
- minimize total cost.

$$Max\ (Min) \sum_{j=1}^{n} b_j xj$$

Constraint examples:

- $\leq$ constraints are typically considered for capacity constraints where you don't want to exceed capacity available.
- $\geq$ constraints are used to model demand requirements where you want to ensure that at least certain level of demand is satisfied.
- $=$ constraints are used when you want to match exactly certain activities with a given requirement. For example, a job position can only be filled with one resource, and you have a set of possible qualified resources to assign to the job.

$$\sum_{j=1}^{n} a_{i,j} x_j \leq (=, \geq)\ K_i\ (i = 1..m)$$

$$x_j \geq 0\ \ (j = 1..n)$$

$$x_j \geq 0\ \ and\ integer\ \ for\ some\ j$$

$$x_j\ \ in\ \{0,1\}\ for\ some\ j$$

Solving MIP Problems

In Mixed Integer linear Programming it is possible to have equivalent formulations of a problem

- But the performance of a MIP solver can be drastically different.
- This is why when formulating a MIP model, its very important to understand how the MIP algorithms behind the MIP solver behave.
- Hence, we will present a limited discussion of the solution process associated with Linear Programming and Mixed Integer Linear Programming.

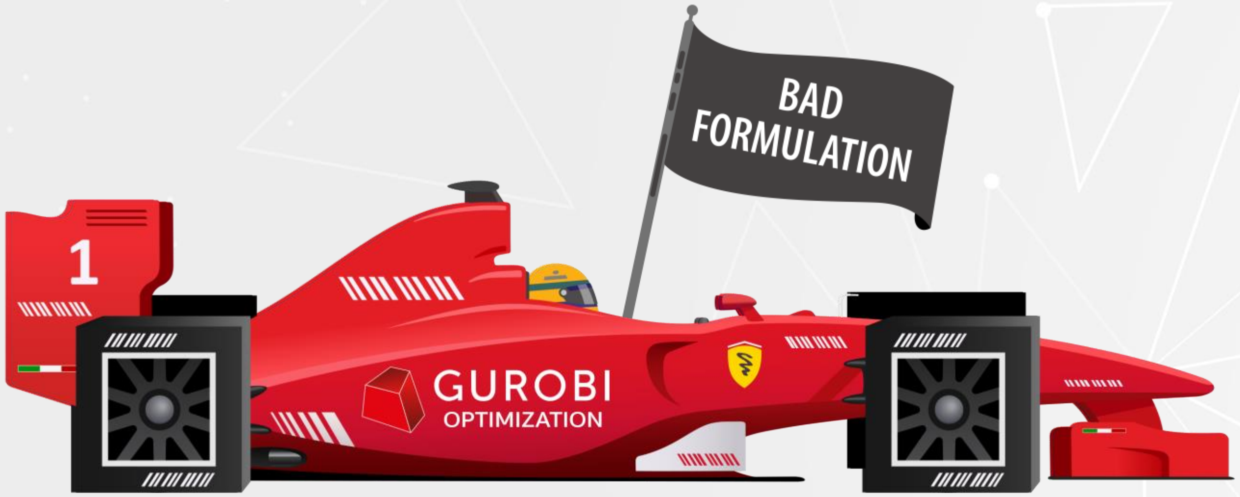
In Mixed Integer linear Programming it is possible to have equivalent formulations of a problem

- But the performance of a MIP solver can be drastically different.
- This is why when formulating a MIP model, its very important to understand how the MIP algorithms behind the MIP solver behave.
- Hence, we will present a limited discussion of the solution process associated with Linear Programming and Mixed Integer Linear Programming.

Solving MIP Problems

In Mixed Integer linear Programming it is possible to have equivalent formulations of a problem

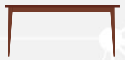
- But the performance of a MIP solver can be drastically different.
- This is why when formulating a MIP model, its very important to understand how the MIP algorithms behind the MIP solver behave.
- Hence, we will present a limited discussion of the solution process associated with Linear Programming and Mixed Integer Linear Programming.



Furniture Factory Problem

Graphical interpretation and solution of an LP problem

LP formulation of furniture problem

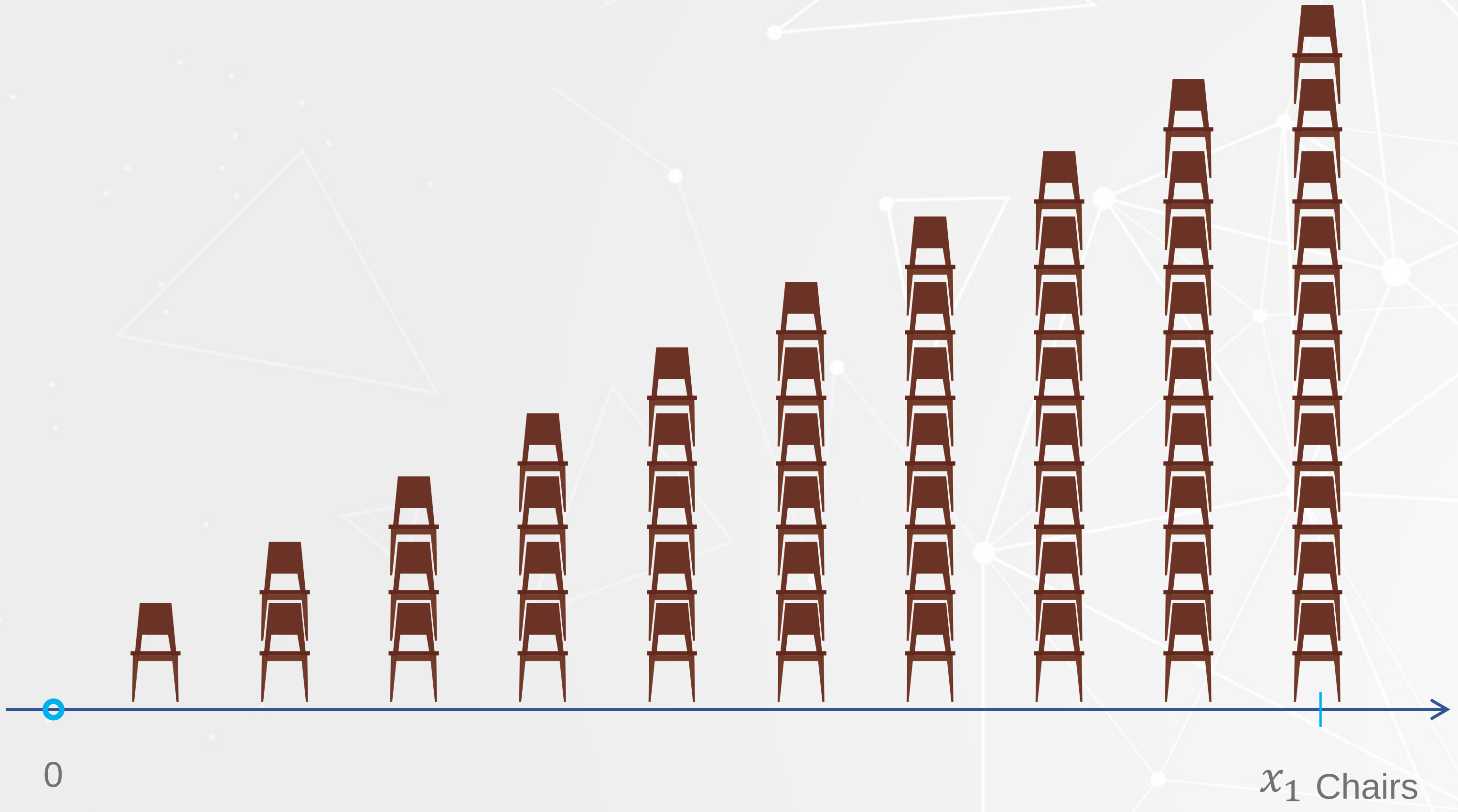
$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400$$ Units of mahogany capacity

$$(3.0).\quad 10x_1 + 15x_2 \leq 450$$ Labor hours capacity

$$\boldsymbol{x_1, x_2 \geq 0}$$ Non – negativity

Graphical solution of Furniture Problem … 1



(1.0). Max revenue $= 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

Graphical solution of Furniture Problem … 2

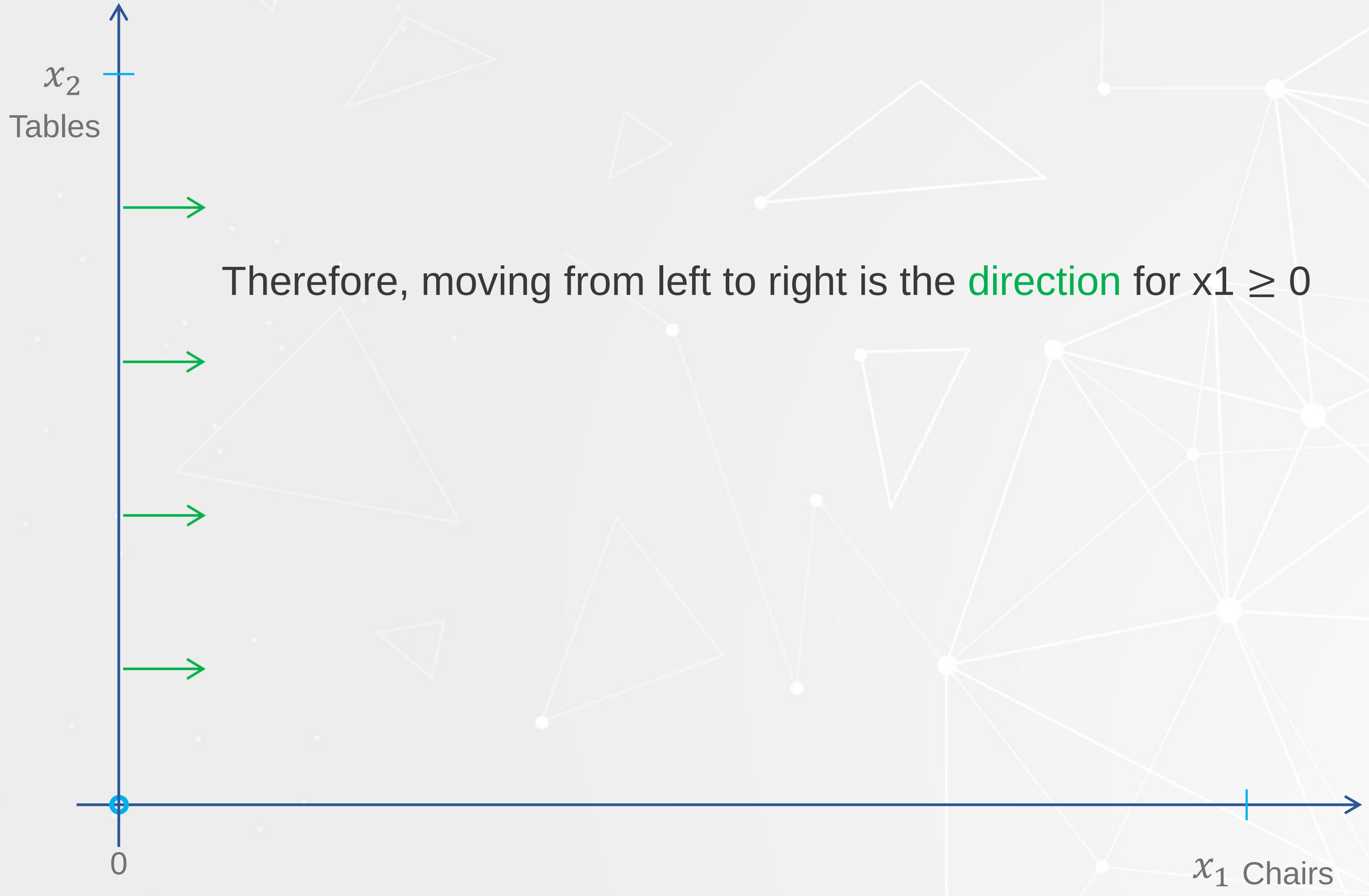
(1.0). Max revenue $= 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

Graphical solution of Furniture Problem … 3



(1.0). Max revenue $= 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

GUROBI OPTIMIZATION

Graphical solution of Furniture Problem … 4

Therefore, moving from low to high is the direction for $x2 \geq 0$

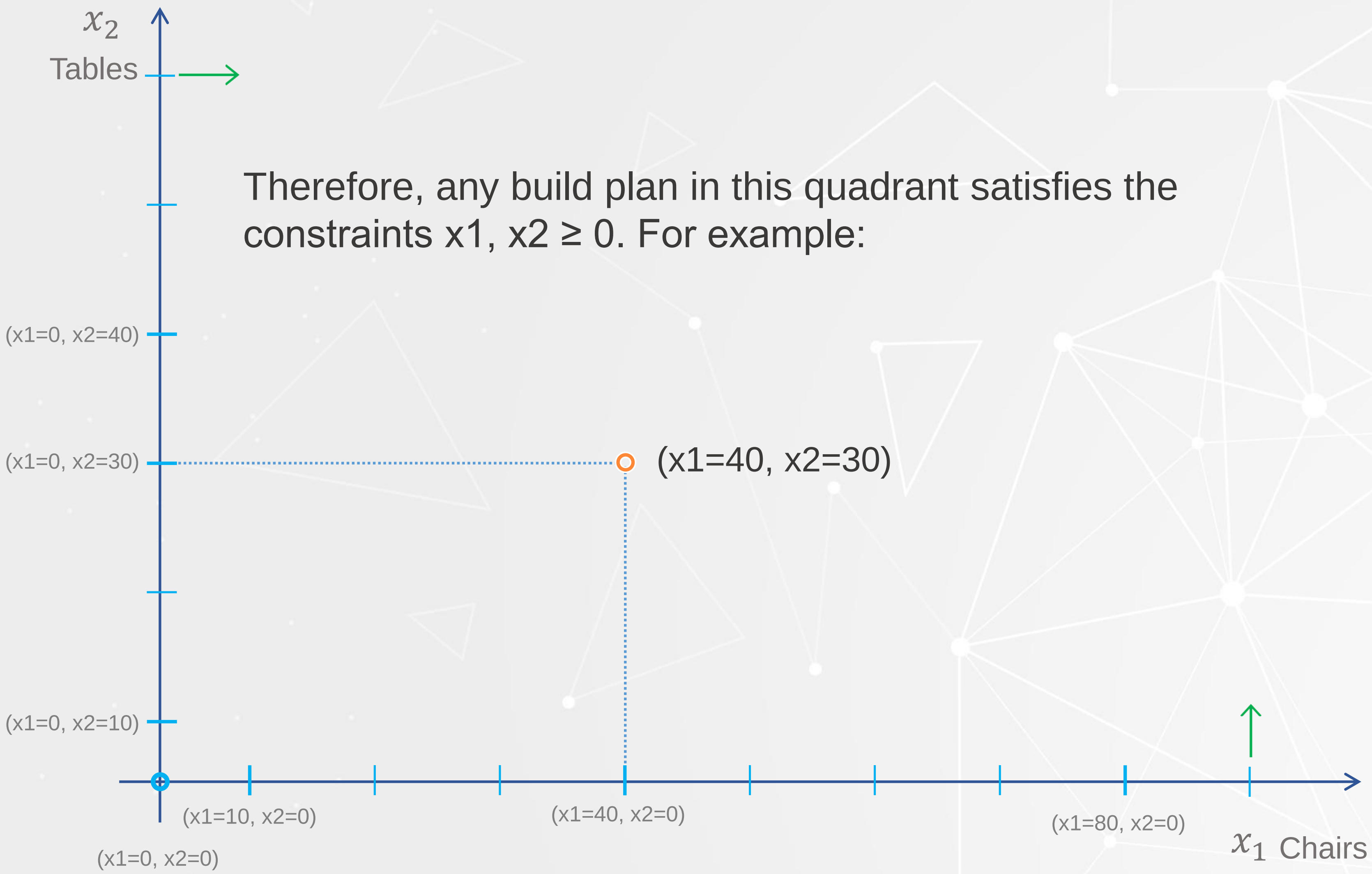
(1.0). Max revenue $= 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non − negativity

Graphical solution of Furniture Problem … 5



Therefore, any build plan in this quadrant satisfies the constraints x1, x2 ≥ 0. For example:

(1.0). $\text{Max revenue} = 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

GUROBI OPTIMIZATION

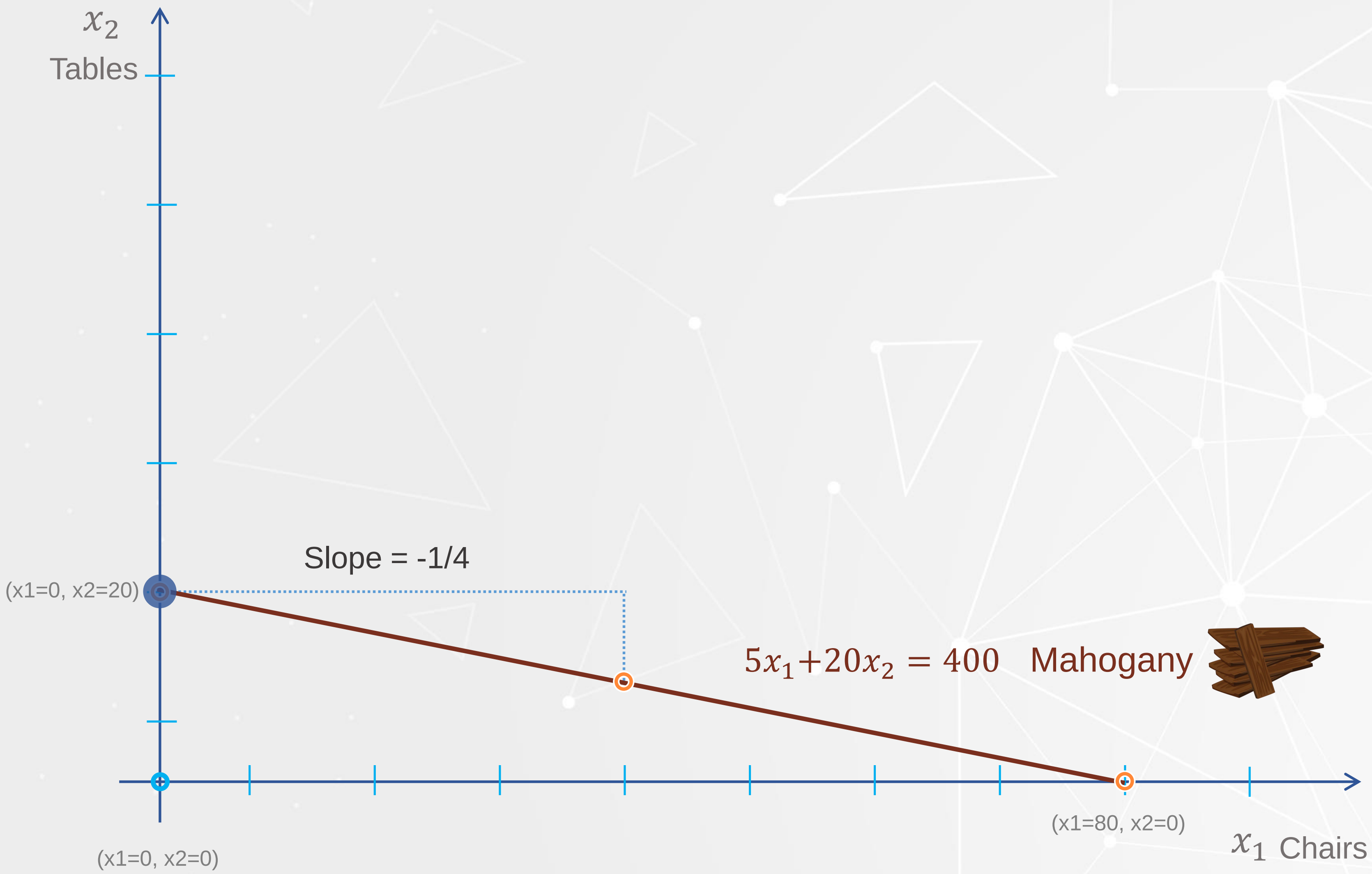
Graphical solution of Furniture Problem … 6

- **$5x1 + 20x2 \leq 400$** (mahogany constraint)
- **$5x1 + 20x2 = 400$** (mahogany equation)

- Expressing x2 in terms of x1
  - **$20x2 = 400 – 5x1$**
  - **$x2 = 400/20 –(5/20)x1$**
  - **Hence, $x2 = 20 –(1/4)x1$**

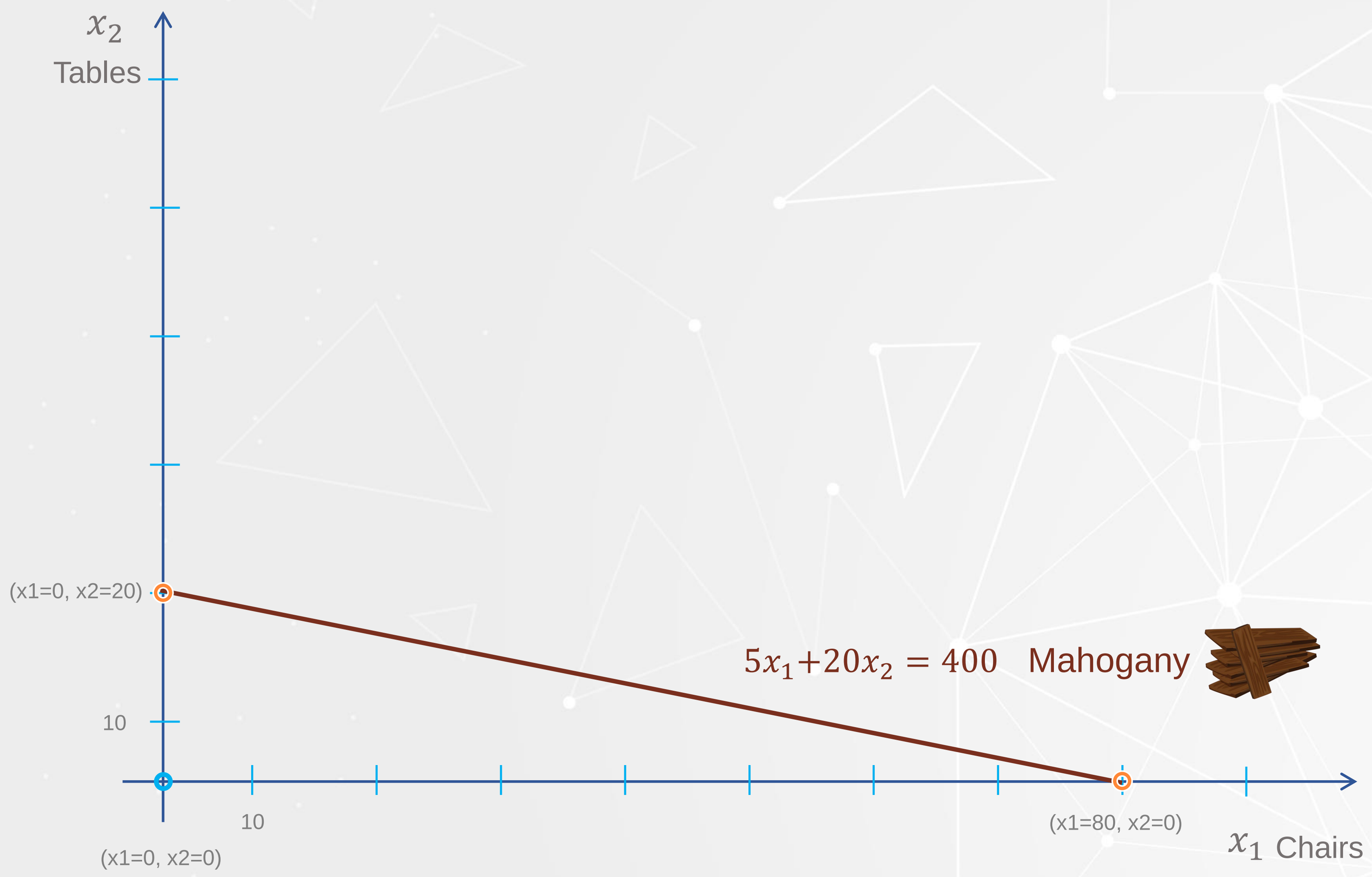
- **If ($x1 = 0$) then ($x2 = 20$)**
- **If ($x1 = 1$) chairs, then $x2 = 20 –(1/4)(x1 = 1) = 19.75$ tables**
- **Mahogany tradeoff tables for chairs is ($1/4 = 0.25$)**

Graphical solution of Furniture Problem … 7



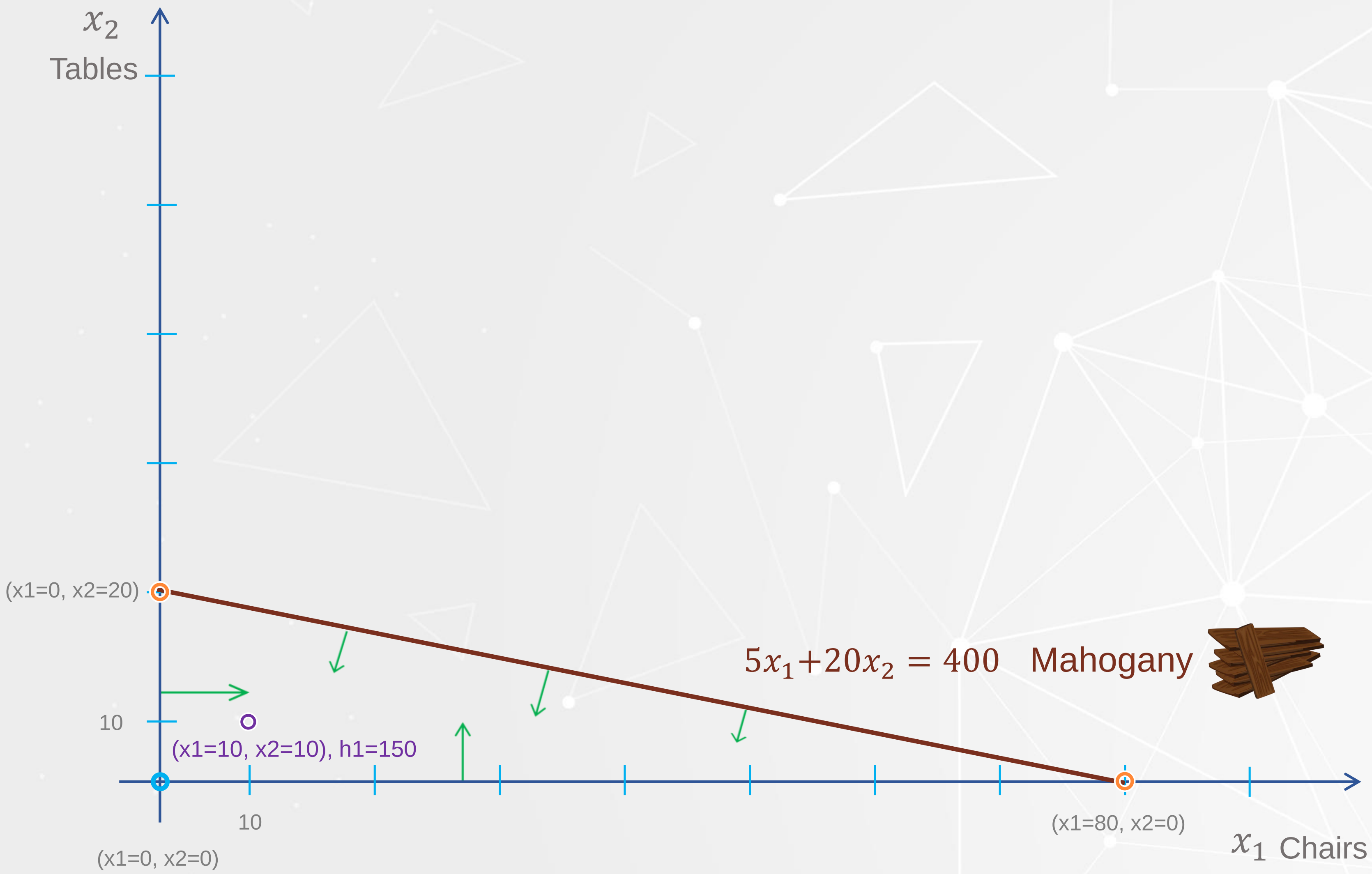
- Let’s graph the equation x2 = 20 –(1/4)x1

Graphical solution of Furniture Problem … 8



- Mahogany constraint: $5x1 + 20x2 \leq 400$.
- (slack variable) h1 ≥ 0: amount of unused mahogany for Production Plan (x1, x2)
- Equation representing mahogany constraint $5x1 + 20x2 + h1 = 400$

Graphical solution of Furniture Problem … 8



- Consider Production Plan (x1=10, x2=10)
- Value of slack variable h1 = 400 – 5(x1=10) – 20(x2=10)=150

Graphical solution of Furniture Problem … 9

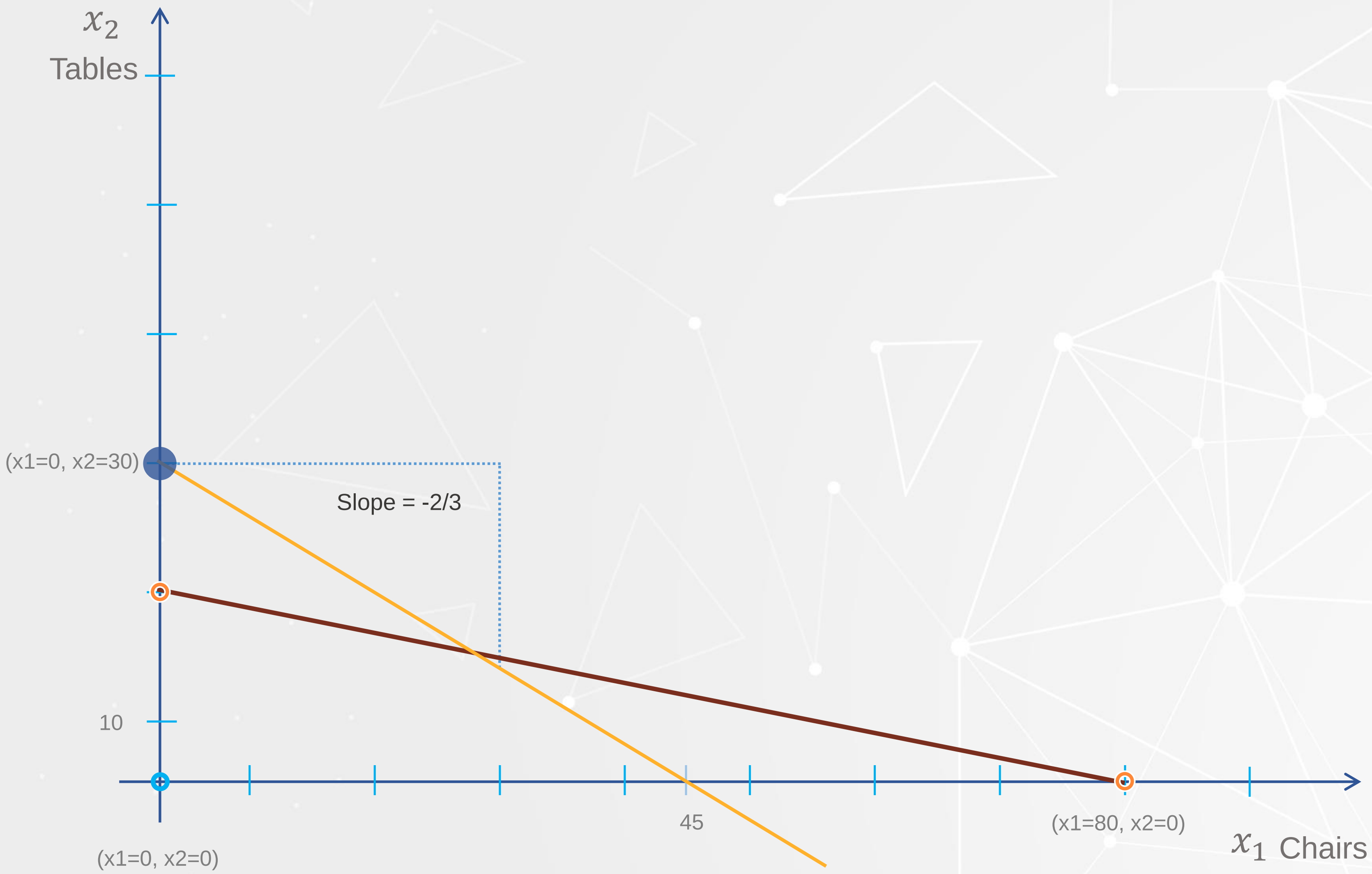
- **$10x1 + 15x2 \le 450$** (labor constraint)
- **$10x1 + 15x2 = 450$** (labor equation)
- Expressing x2 in terms of x1
  - **$x2 = 30 – (2/3)x1$**

- **If (x1 = 0) then (x2 = 30)**

- **If (x1 = 1) chair, then x2 = 30 –(2/3)(x1 = 1) = 29.333 tables**

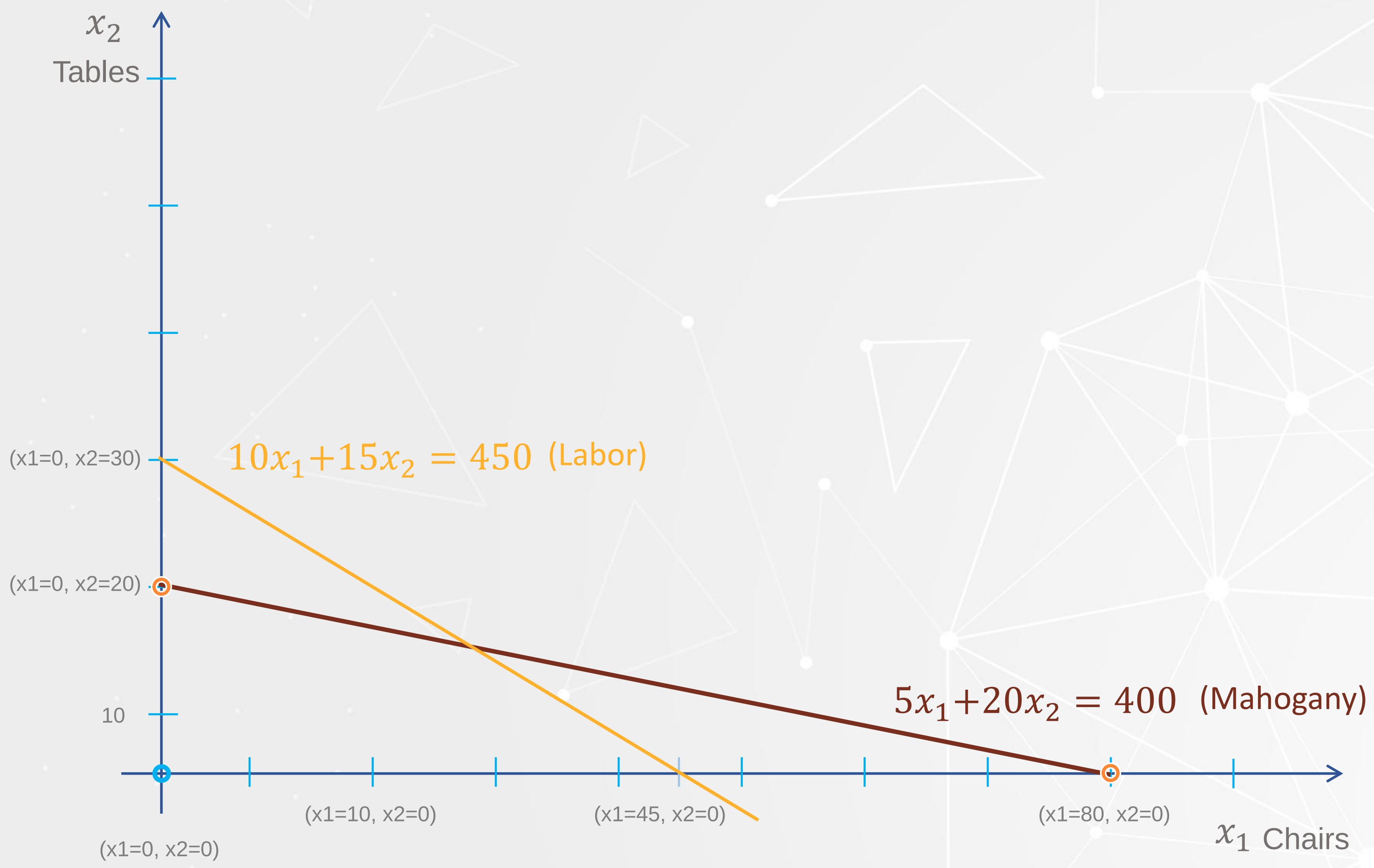
- **Labor tradeoff tables for chairs is (2/3 = 0.667)**

Graphical solution of Furniture Problem … 10



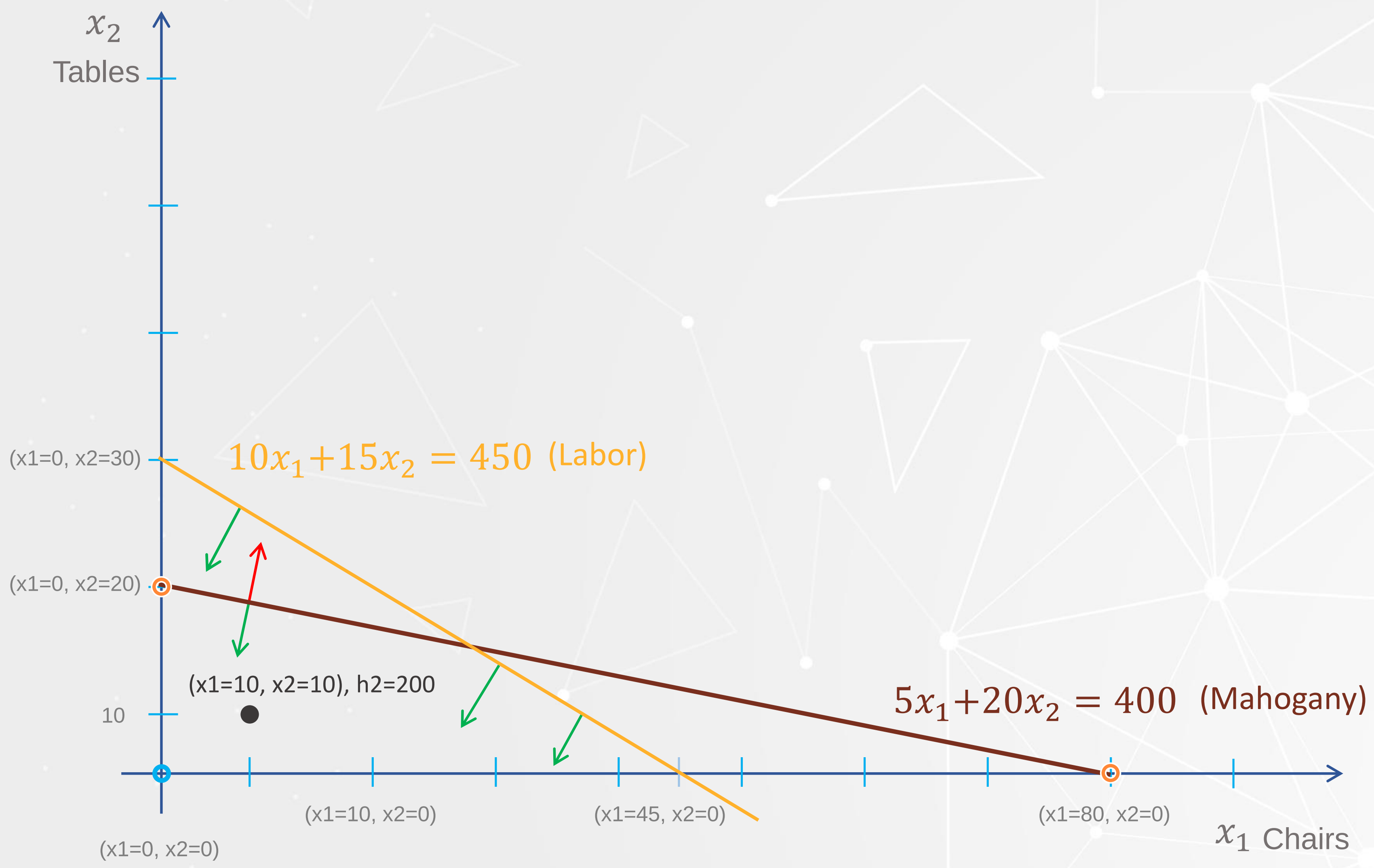
- Let’s graph the equation $x2 = 30 - (2/3)x1$

Graphical solution of Furniture Problem … 11



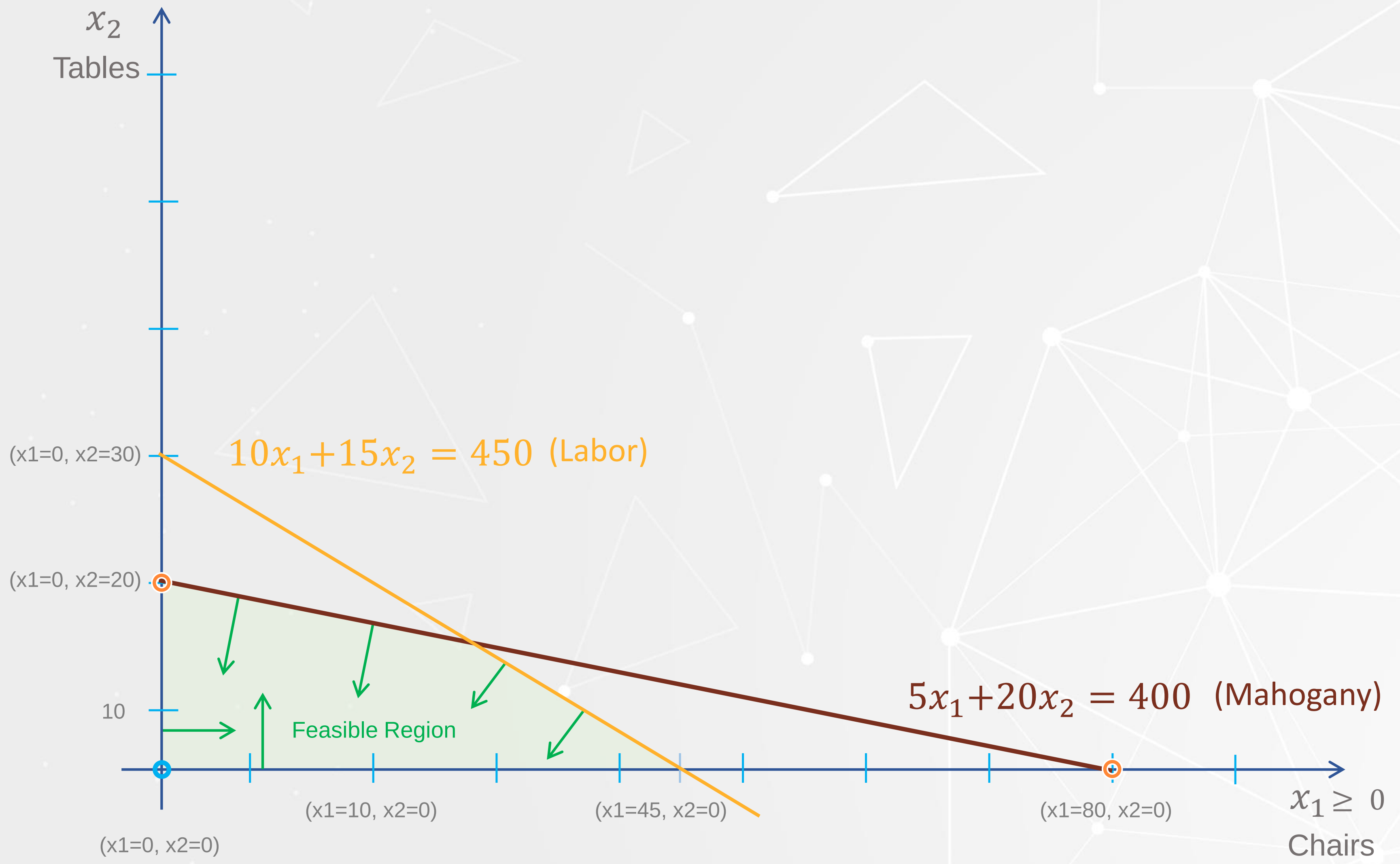
- Labor constraint
  10x1 + 15x2 ≤ 450
- (slack variable) h2 ≥ 0: amount of unused labor for production plan (x1, x2)
- Equation representing labor constraint
  10x1 + 15x2 + h2 = 450

Graphical solution of Furniture Problem … 11



- Production Plan (x1=10, x2=10)
- Slack variable value h2 = 450 – 10(x1=10) – 15(x2=10) = 200

Graphical solution of Furniture Problem … 12

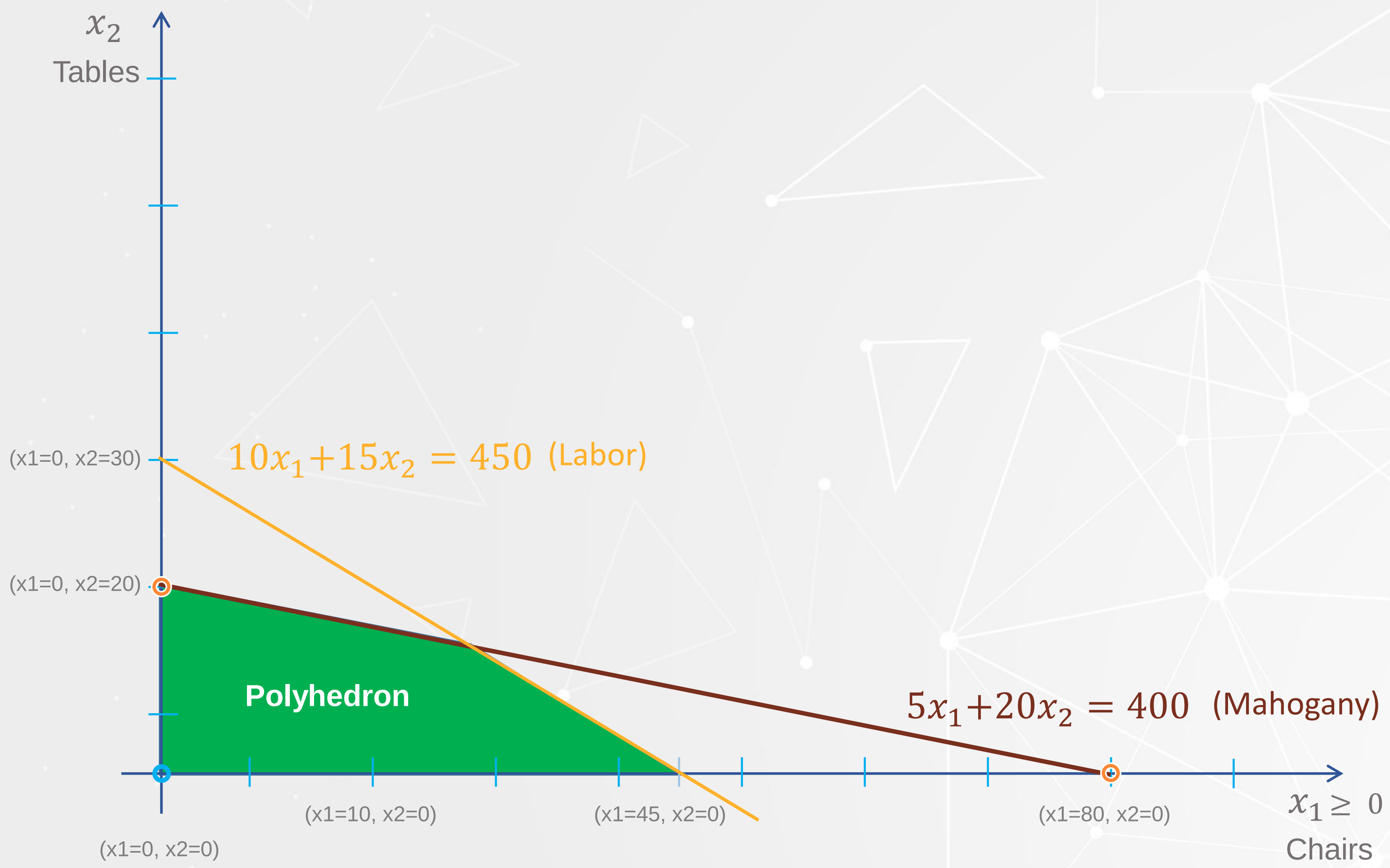


(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

Graphical solution of Furniture Problem … 13



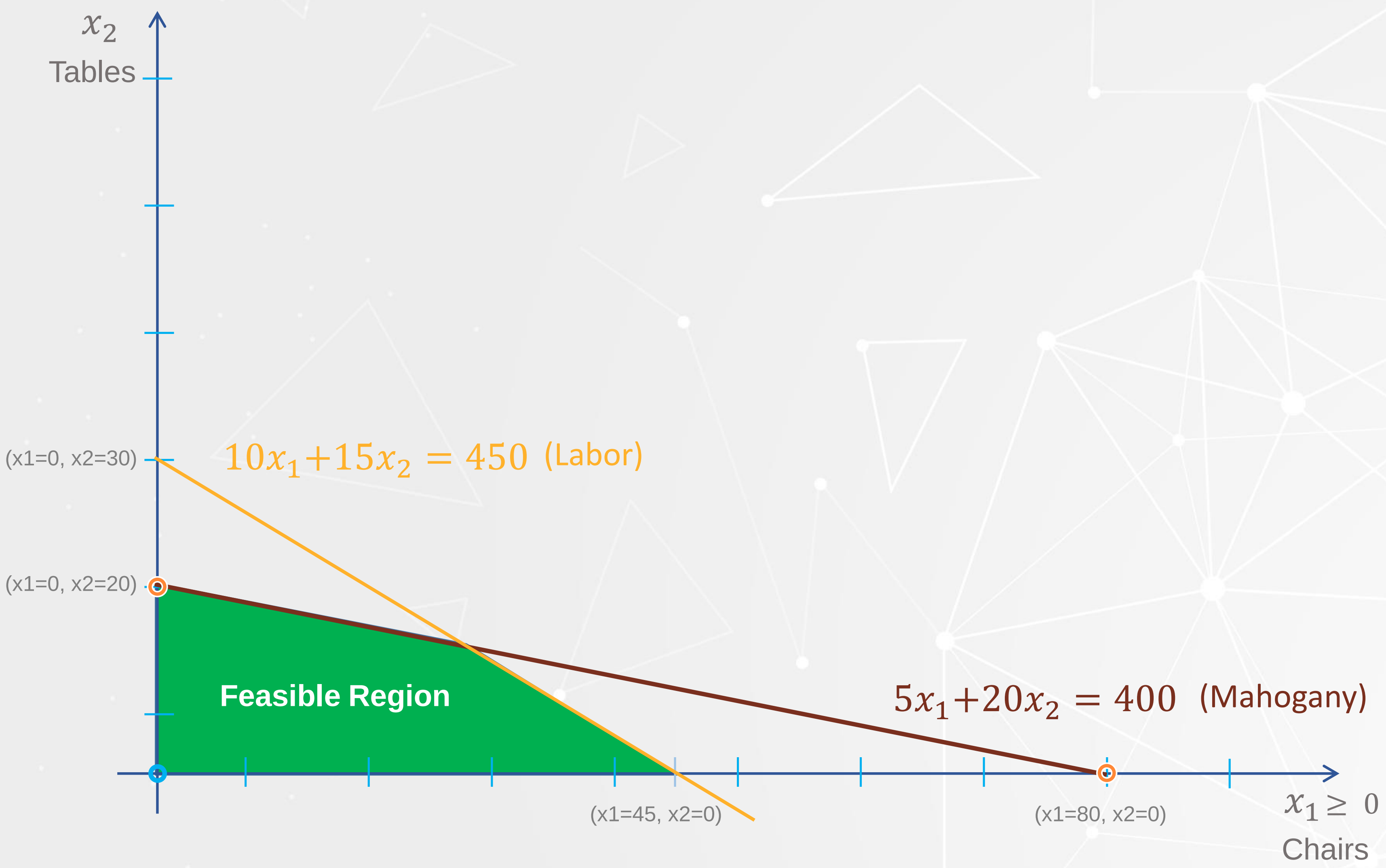
(2.0) $5x_1 + 20x_2 \leq 400$ Units of mahogany capacity

(3.0). $10x_1 + 15x_2 \leq 450$ Labor hours capacity

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

- In the theory of linear programming the feasible region is called a polyhedron.

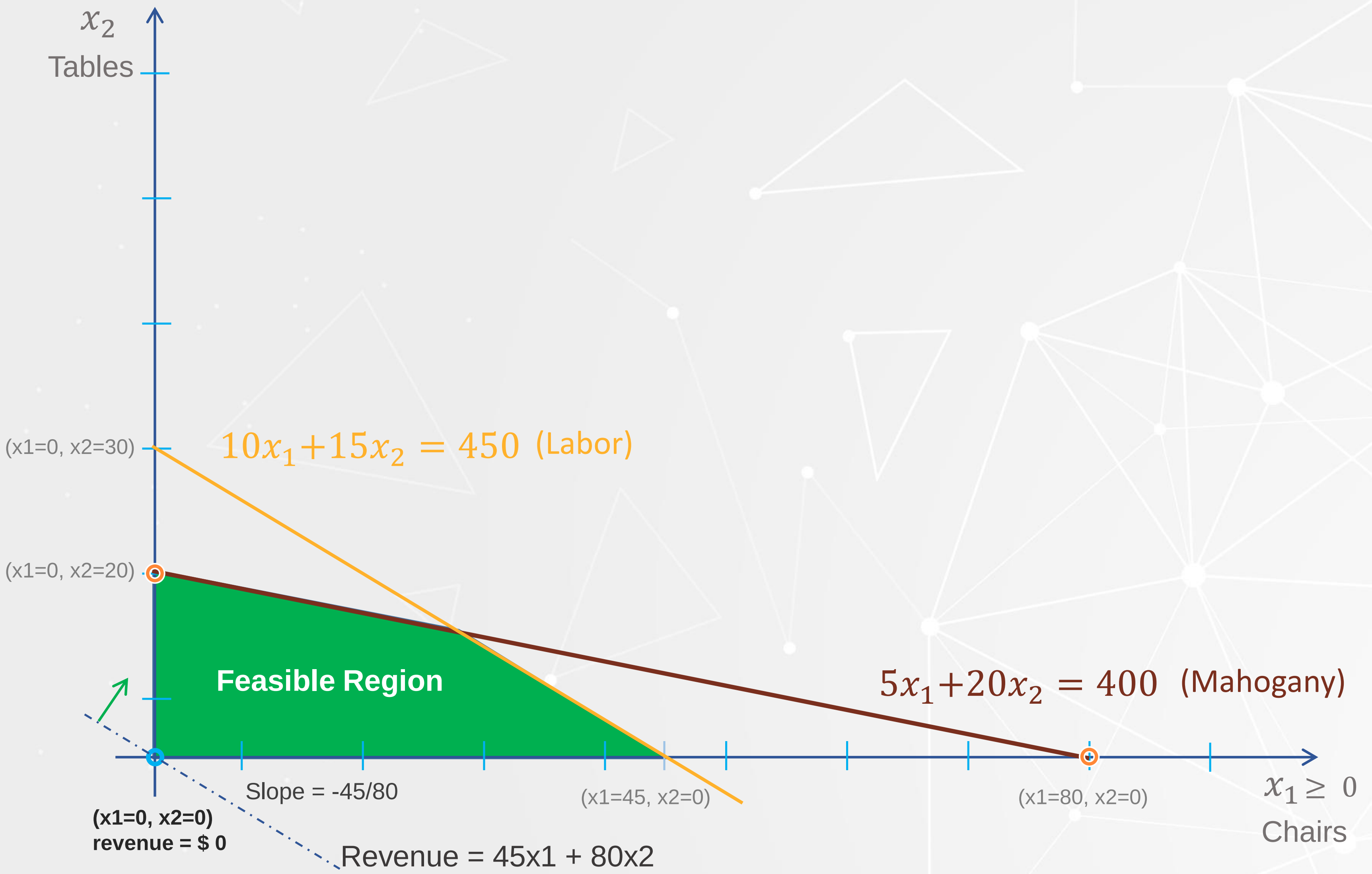
Graphical solution of Furniture Problem … 14



- The objective function: revenue = 45x1 + 80x2

  x2 = revenue/80 –(45/80)x1

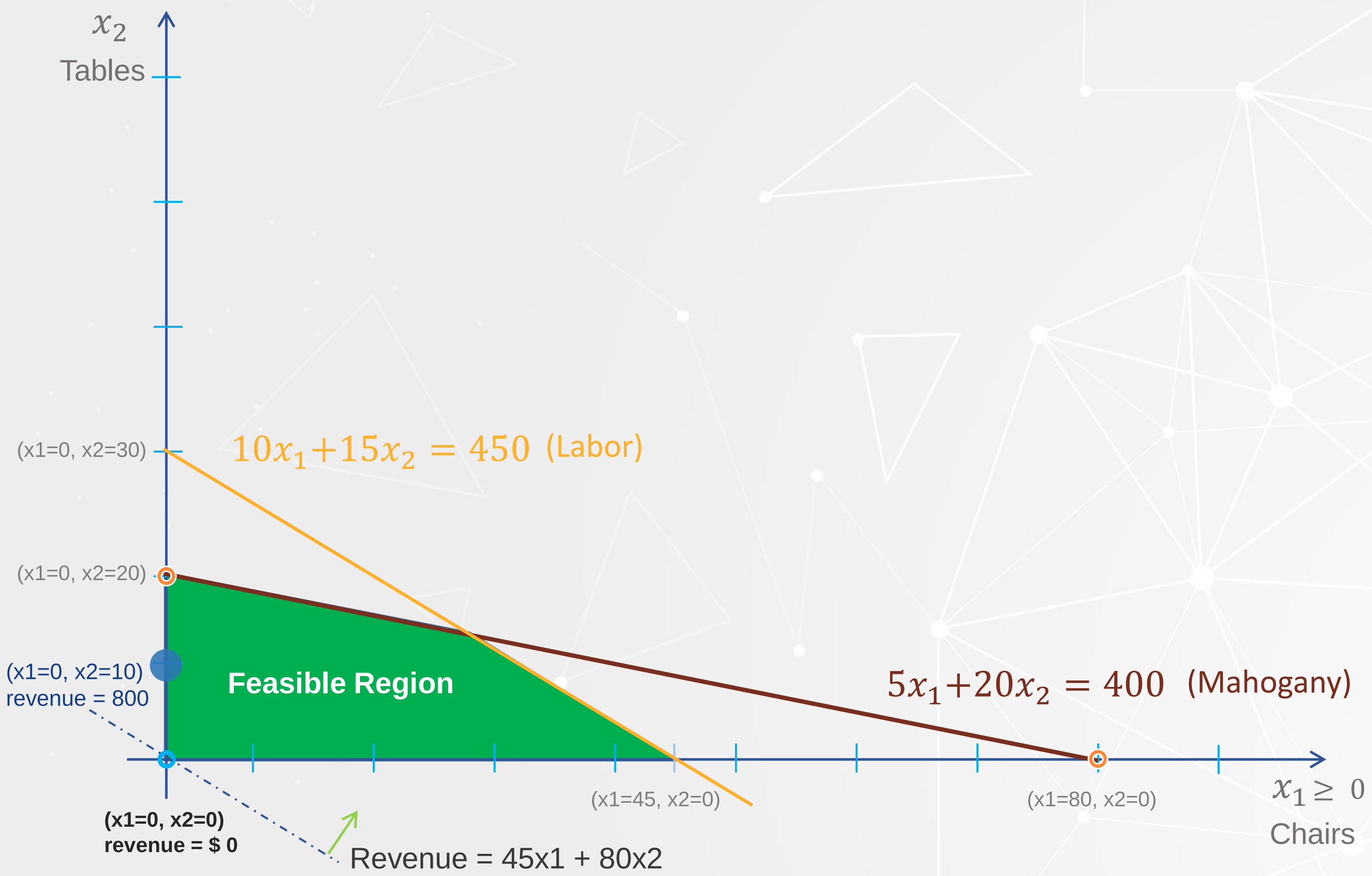
Graphical solution of Furniture Problem … 14



x2 =
revenue/80 –(45/80)x1

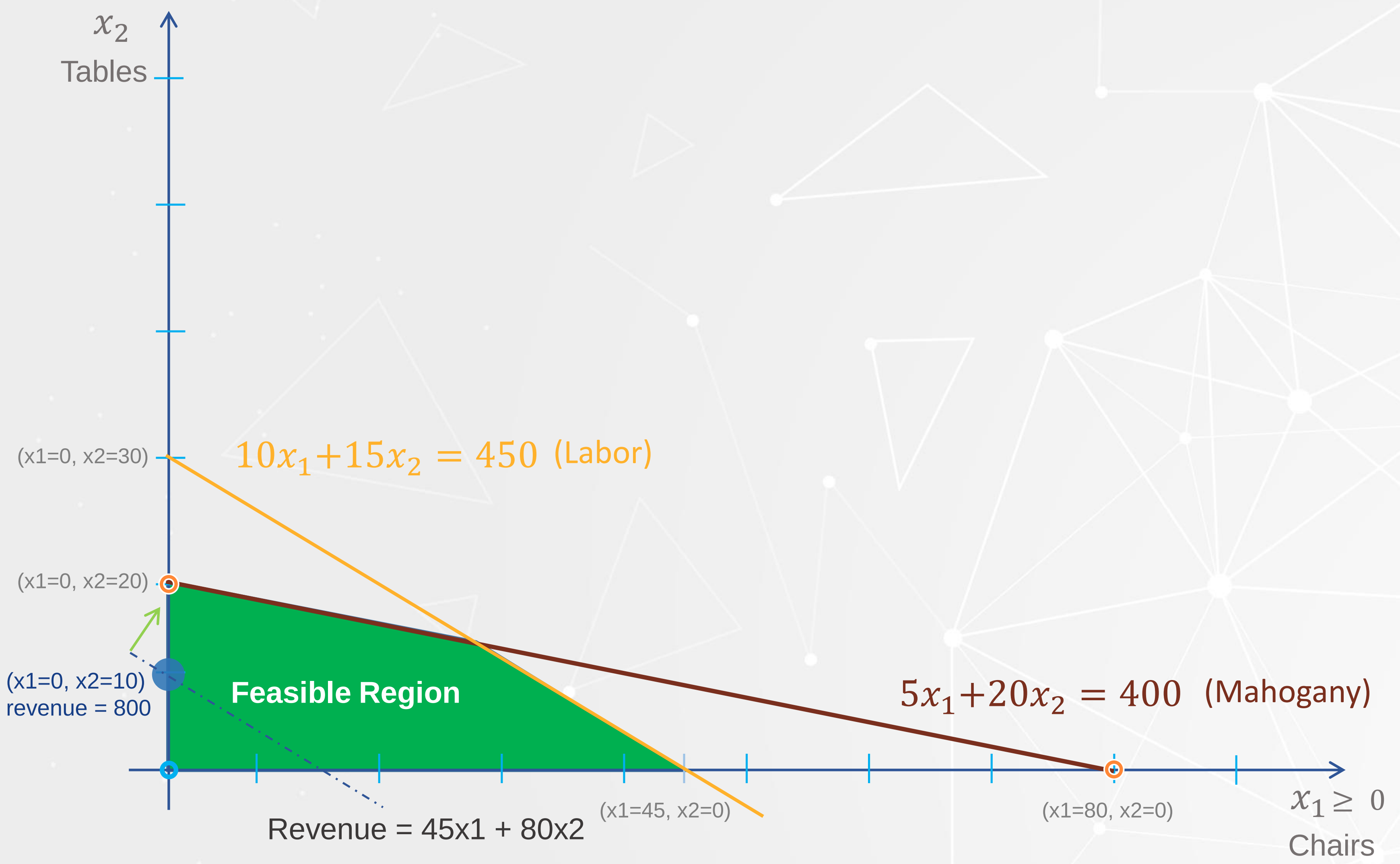
- If (x1=0, x2=0), then revenue is $0.00.

Graphical solution of Furniture Problem … 15



- Production Plan
  (x1 = 0, x2 = 10)
- Generates a revenue
  = 45(x1=0) + 80(x2=10)
  = $800

Graphical solution of Furniture Problem … 15



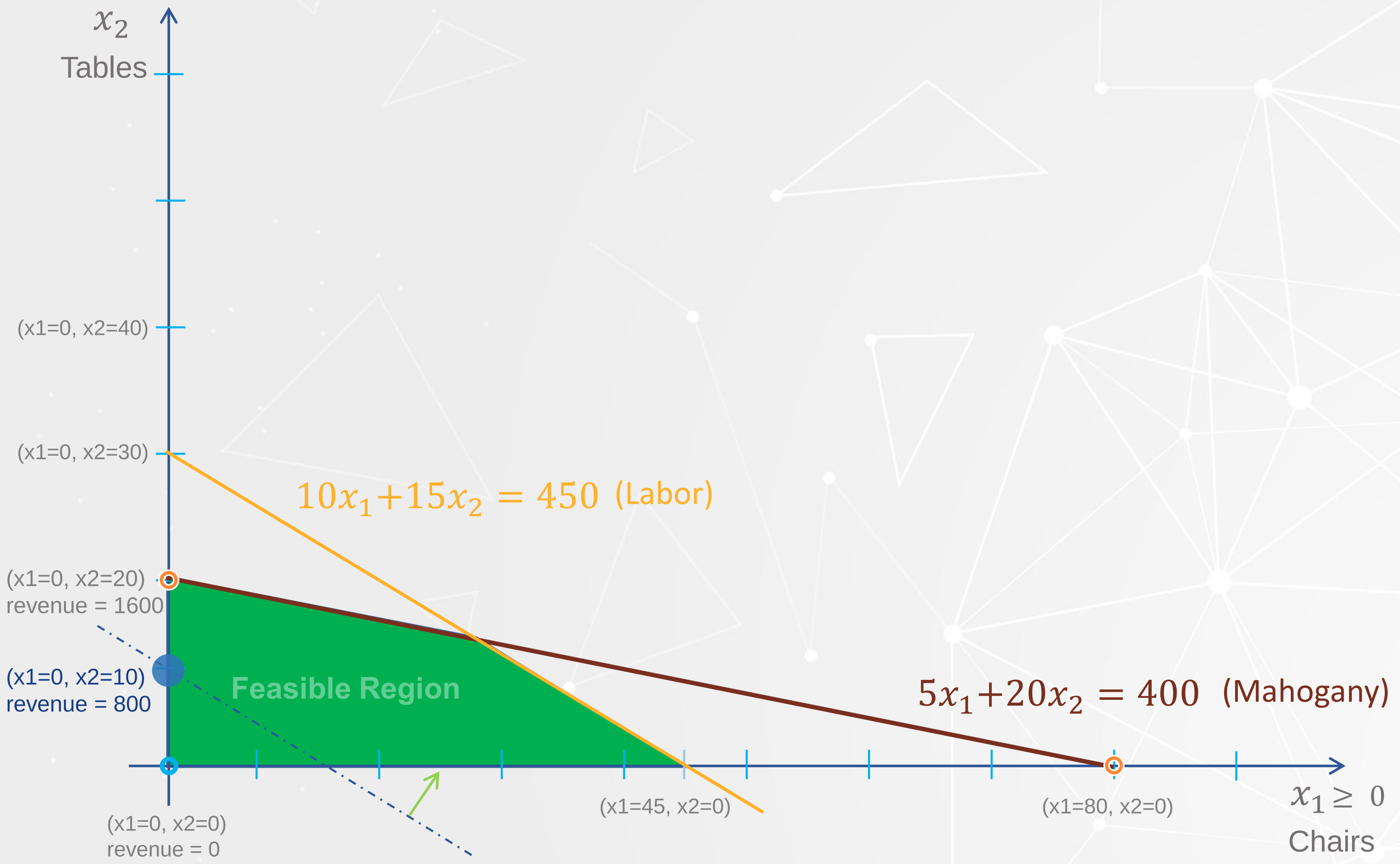
- Mahogany slack variable:
  h1 =
  400 – 5(x1=0) –
  20(x2=10)=200

- Labor slack variable:
  h2 =
  450 – 10(x1=0) –
  15(x2=10) = 150

- 200 units of unused mahogany capacity

- 150 units of unused labor capacity

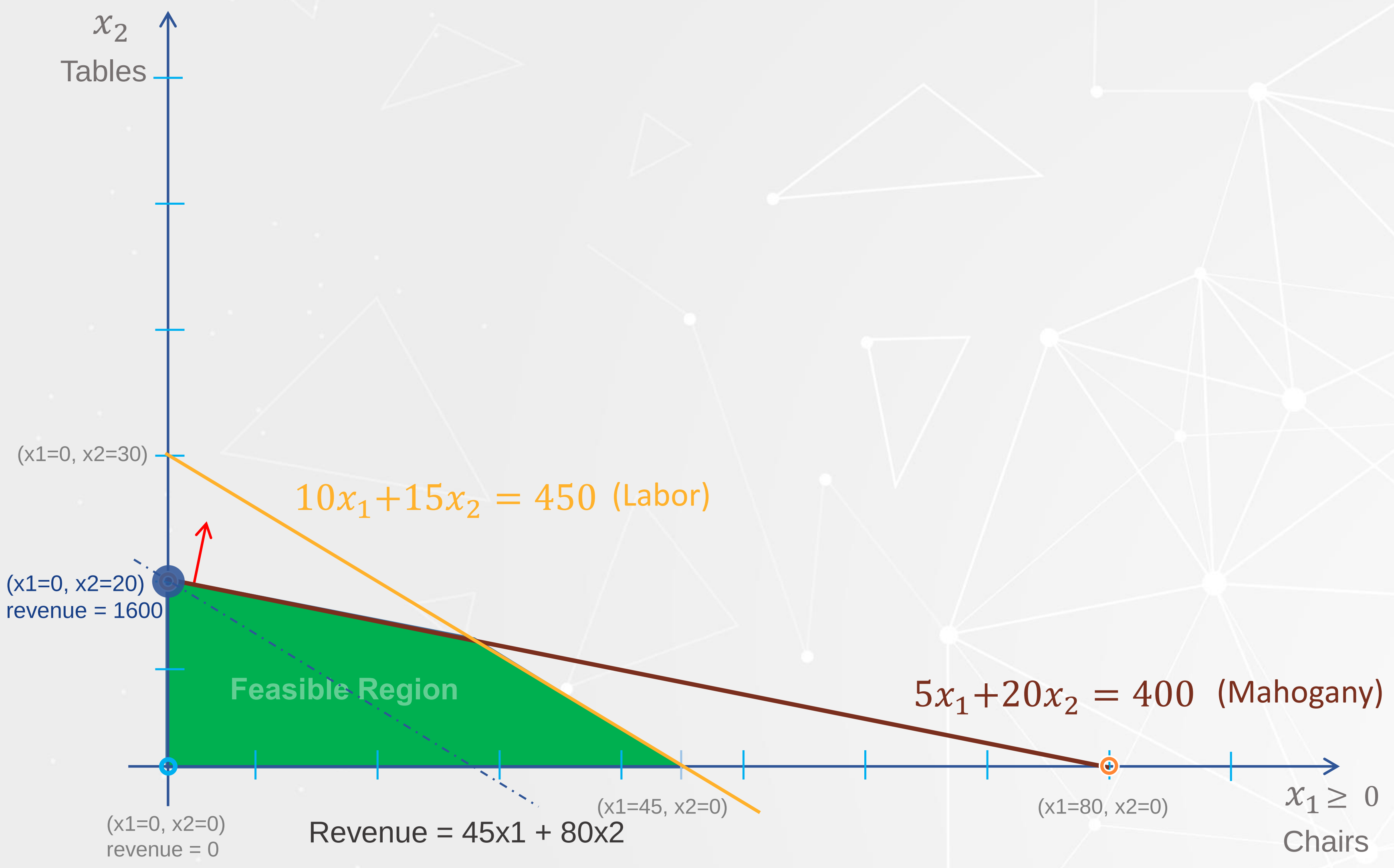
Graphical solution of Furniture Problem … 16



- How far can we increase the production of tables?
- Production Plan (x1=0, x2=20)

  Revenue =
  45(x1 = 0) + 80(x2=20) =
  $1,600

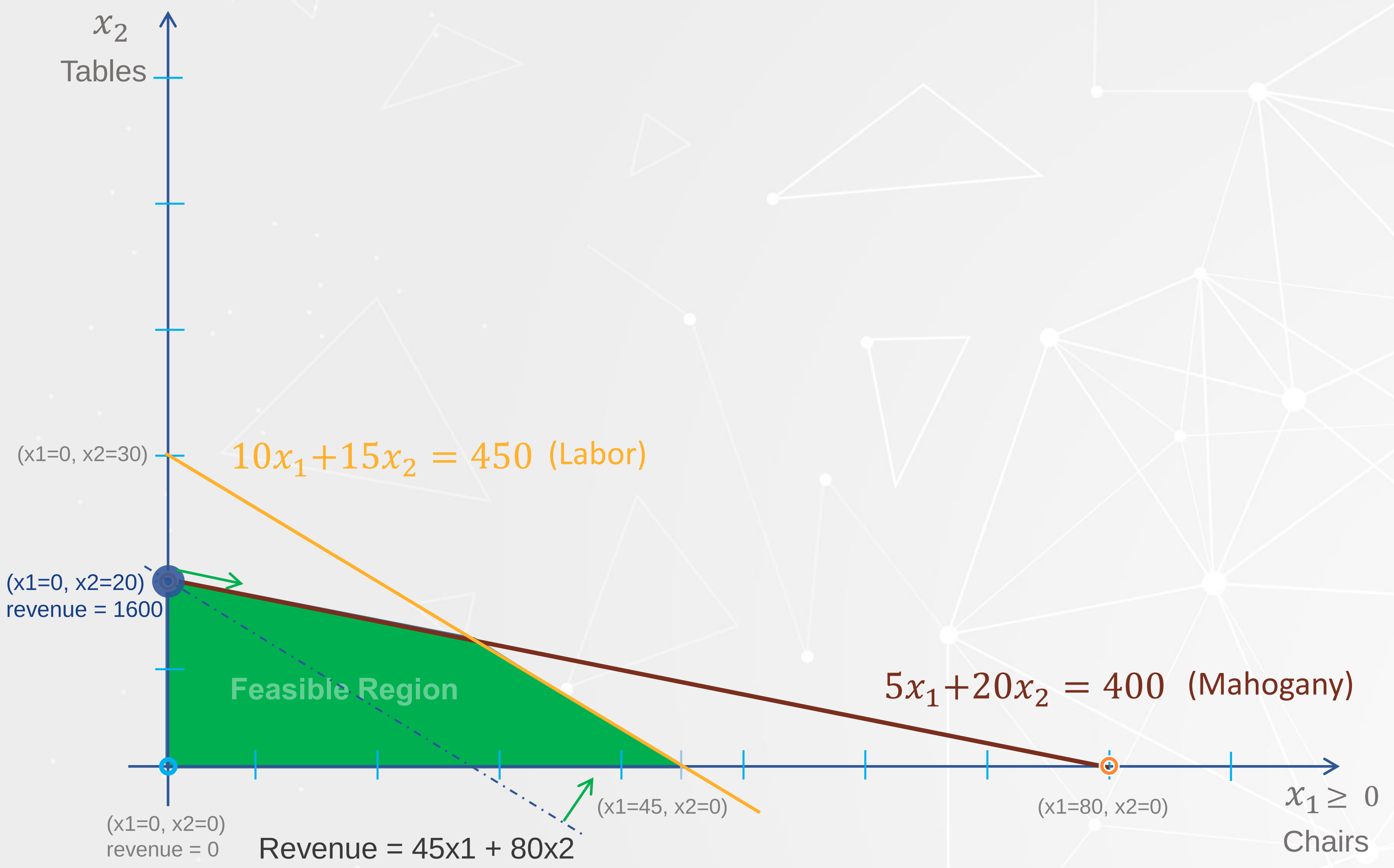
Graphical solution of Furniture Problem … 16



- Can we continue increasing the production tables?
- Production plan (x1=0, x2=21)

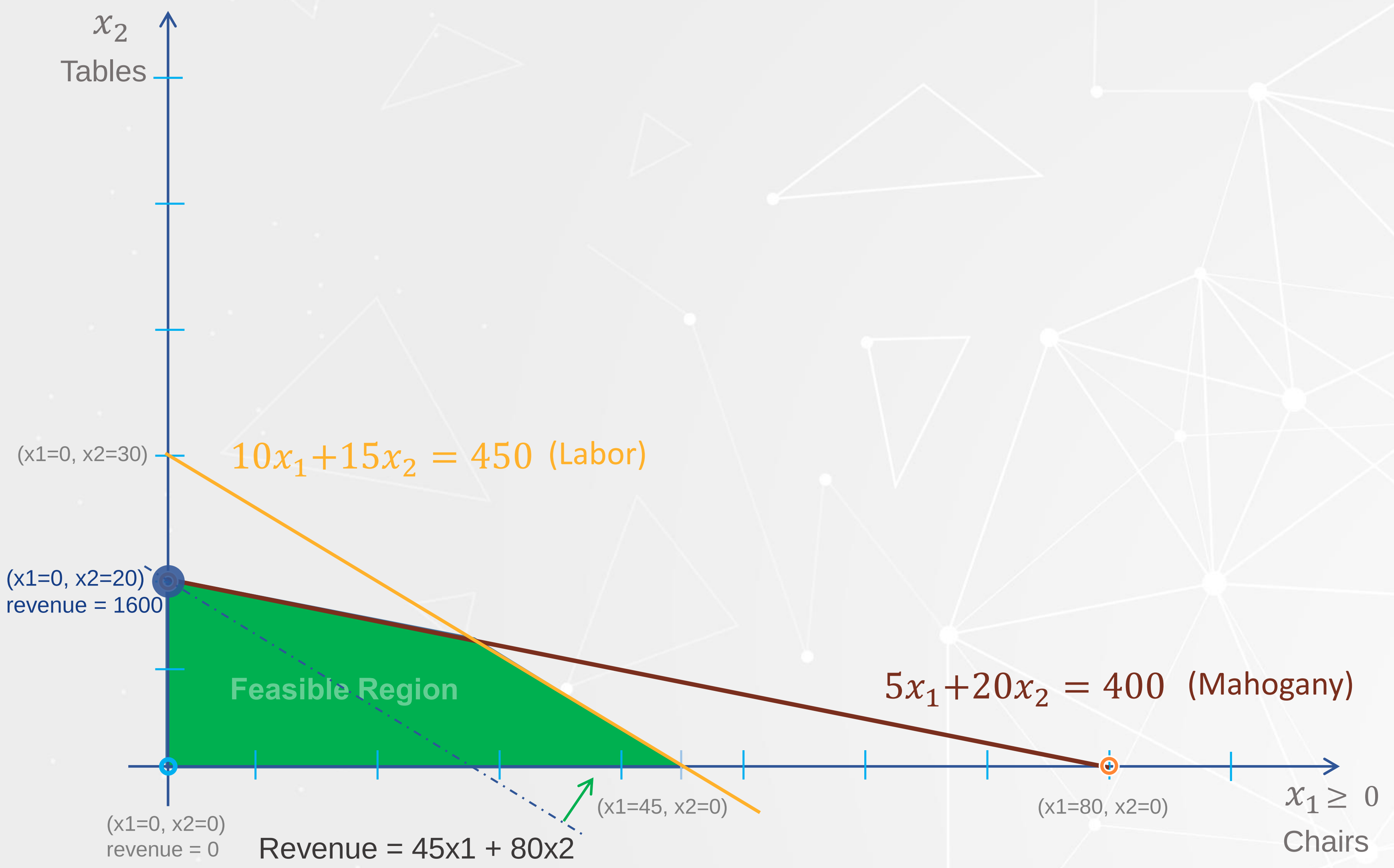
  Revenue =
  45(x1 = 0) + 80(x2=21) = $1,680
- Mahogany slack variable h1 = 400 – 5(x1=0) – 20(x2=21)= -20 !!!
  Infeasible

Graphical solution of Furniture Problem … 17



- What else we can do?

Graphical solution of Furniture Problem … 17

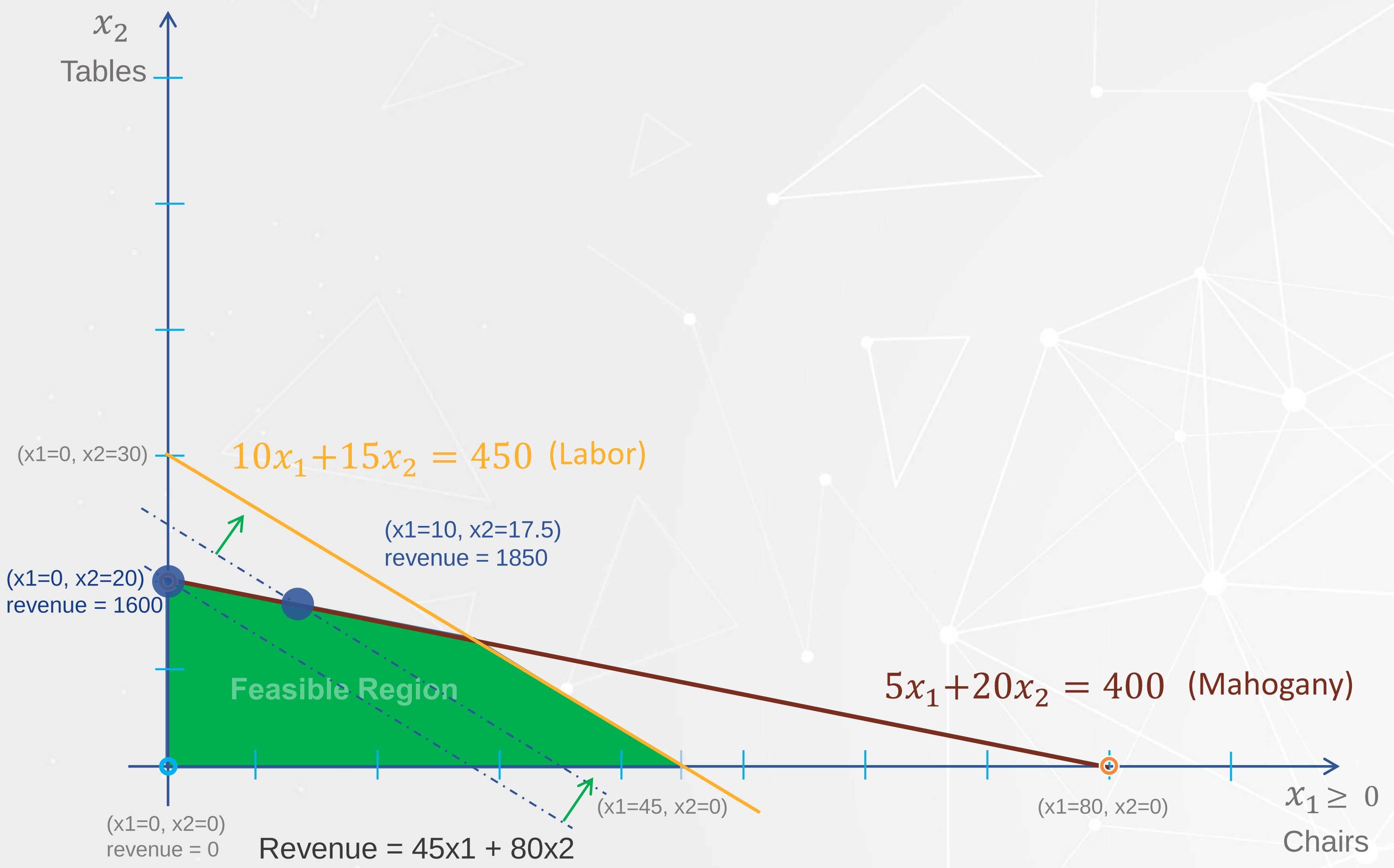


- Mahogany equation

  x2 =

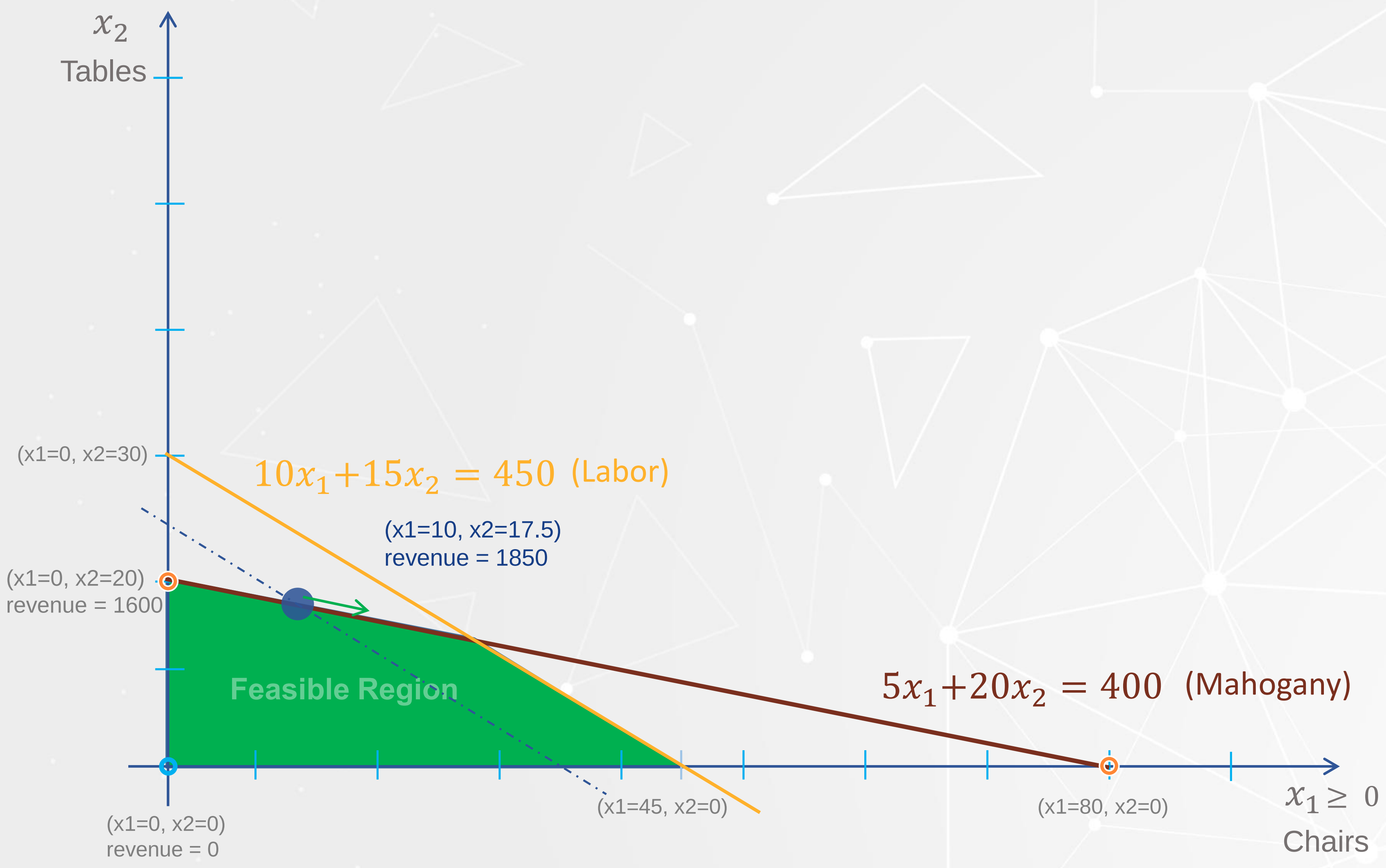
  20 –(1/4)x1

Graphical solution of Furniture Problem … 18



- How much can we keep increasing the production of chairs while keeping the production of tables as high as we can?
- If we build 10 chairs, then: x2 = 20 –(1/4)(x1=10) = 17.5 tables.
- Mahogany slack variable h1 = 400 – 5(x1=10) – 20(x2=17.5) = 0

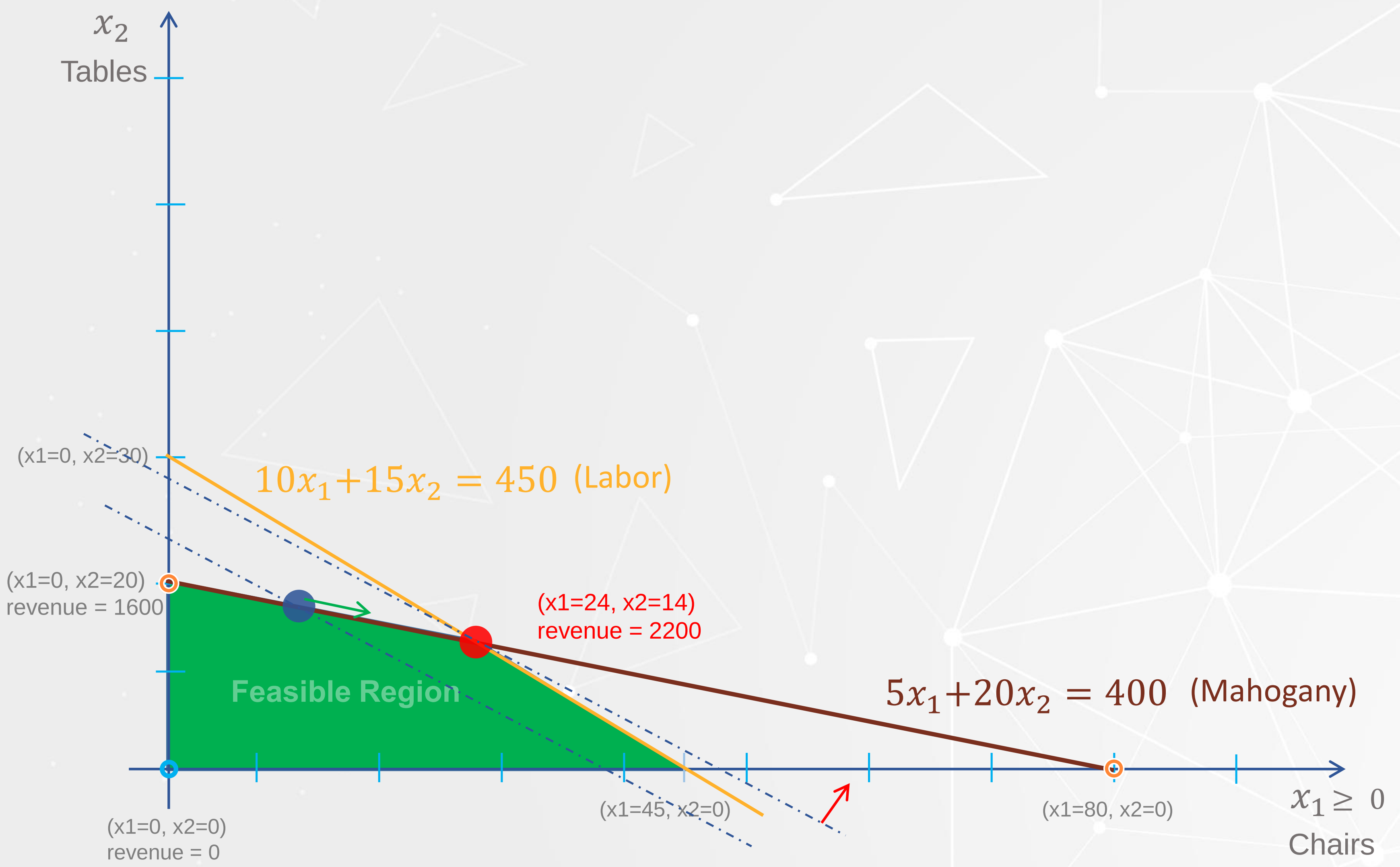
Graphical solution of Furniture Problem … 18



- Production plan (x1=10, x2=17.5)
- Revenue = 45(x1=10) + 80(x2=17.5) = $1,850
- Labor slack variable h2 = 450 – 10(x1=10) – 15(x2=17.5) = 87.5

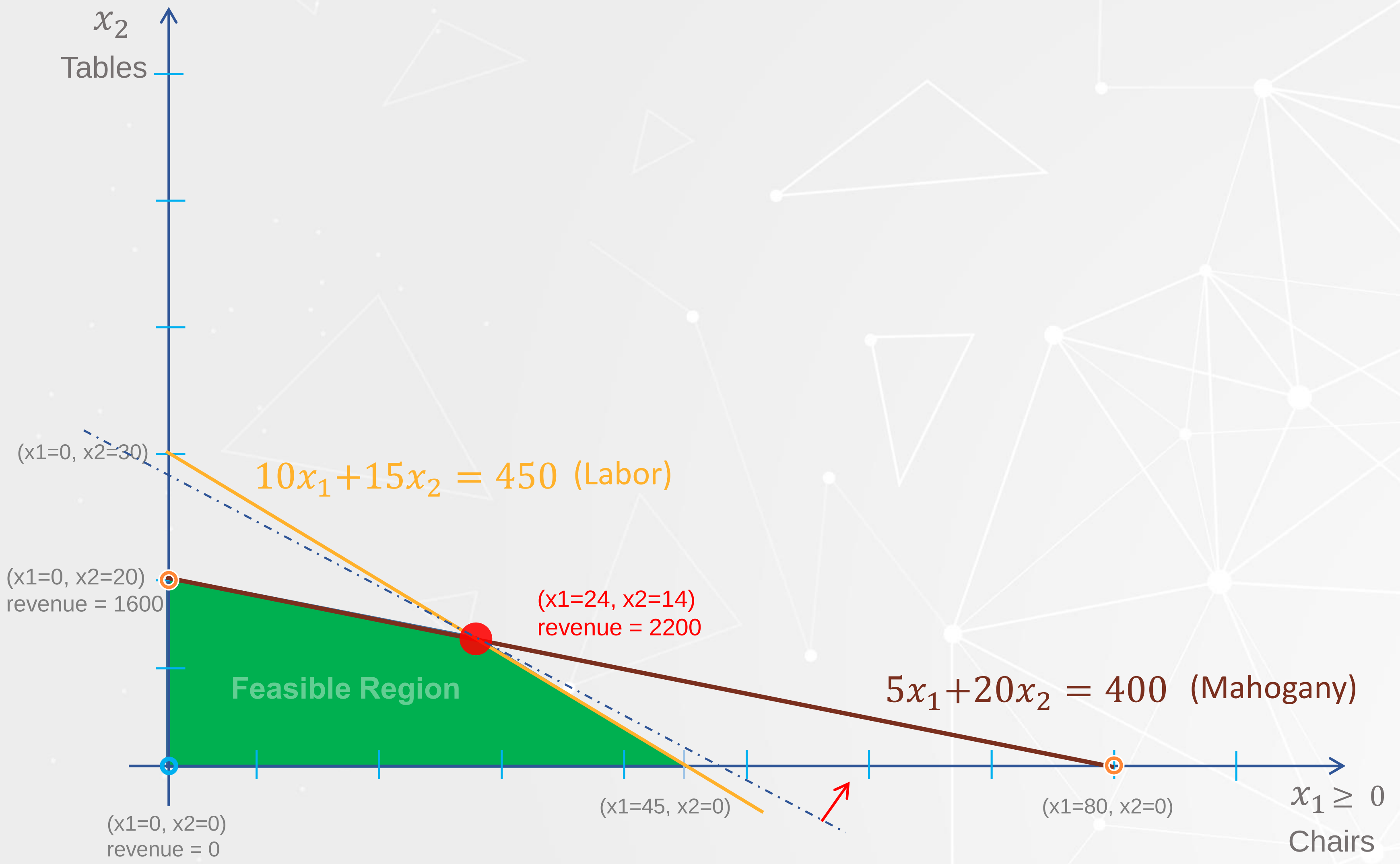
GUROBI OPTIMIZATION

Graphical solution of Furniture Problem … 19



- Observe that when the production plan moving along the mahogany equation hits the labor equation, we cannot move any further.
- This happens when the equation that defines the mahogany constraint intersects with the equation that defines the labor constraint. The associated production plan is found by solving these system of equations.
- **STOP**, labor constraint limits production plan.

Graphical solution of Furniture Problem … 19



Mahogany

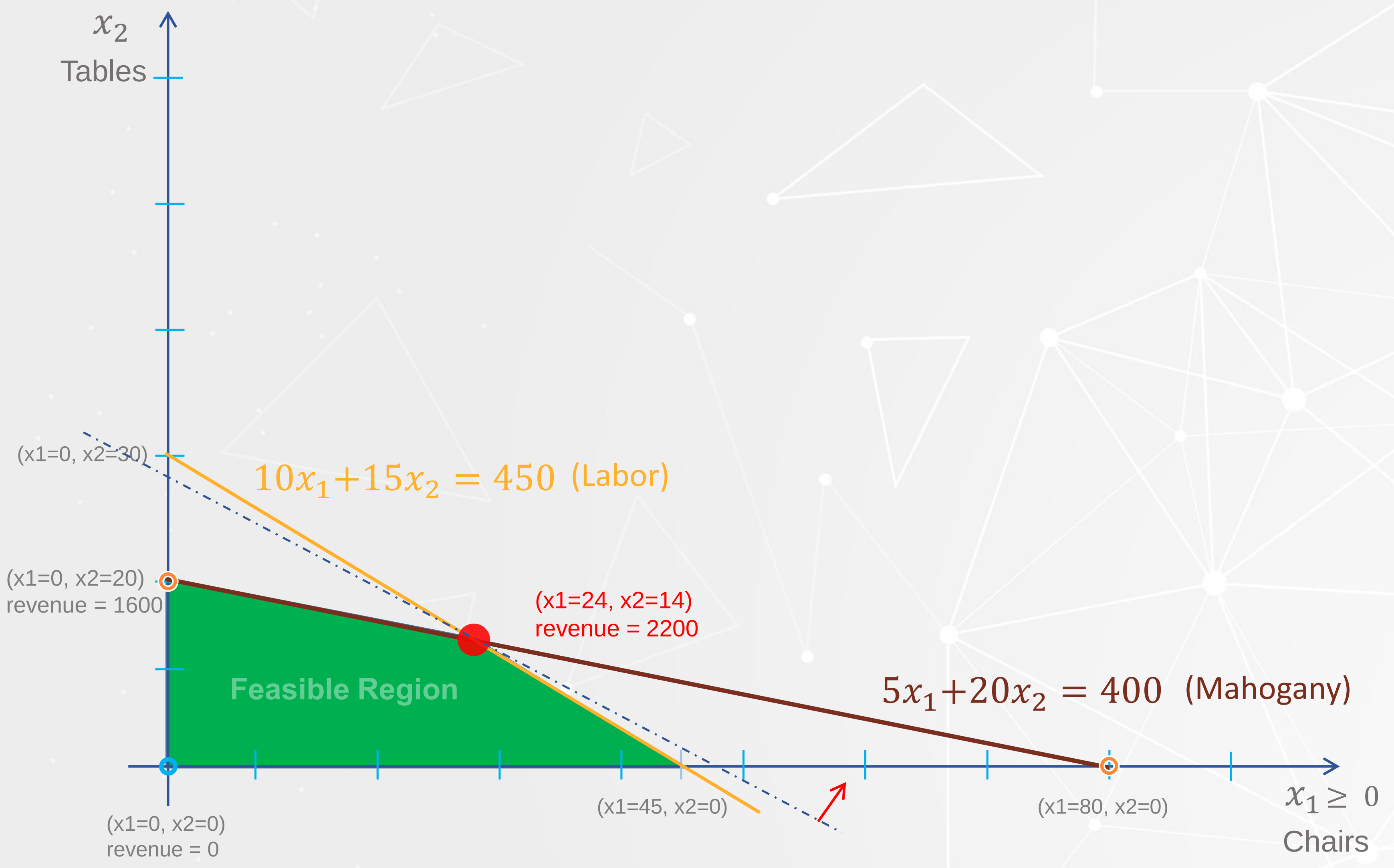
$$5x_1+20x_2 = 400$$

Labor

$$10x_1+15x_2 = 450$$

- Production Plan:
  - 24 chairs
  - 14 tables.
- Revenue = 45(x1=24) + 80(x2=14) = $2,200

Graphical solution of Furniture Problem … 19



Production Plan (x1 =24, x2=14) is optimal

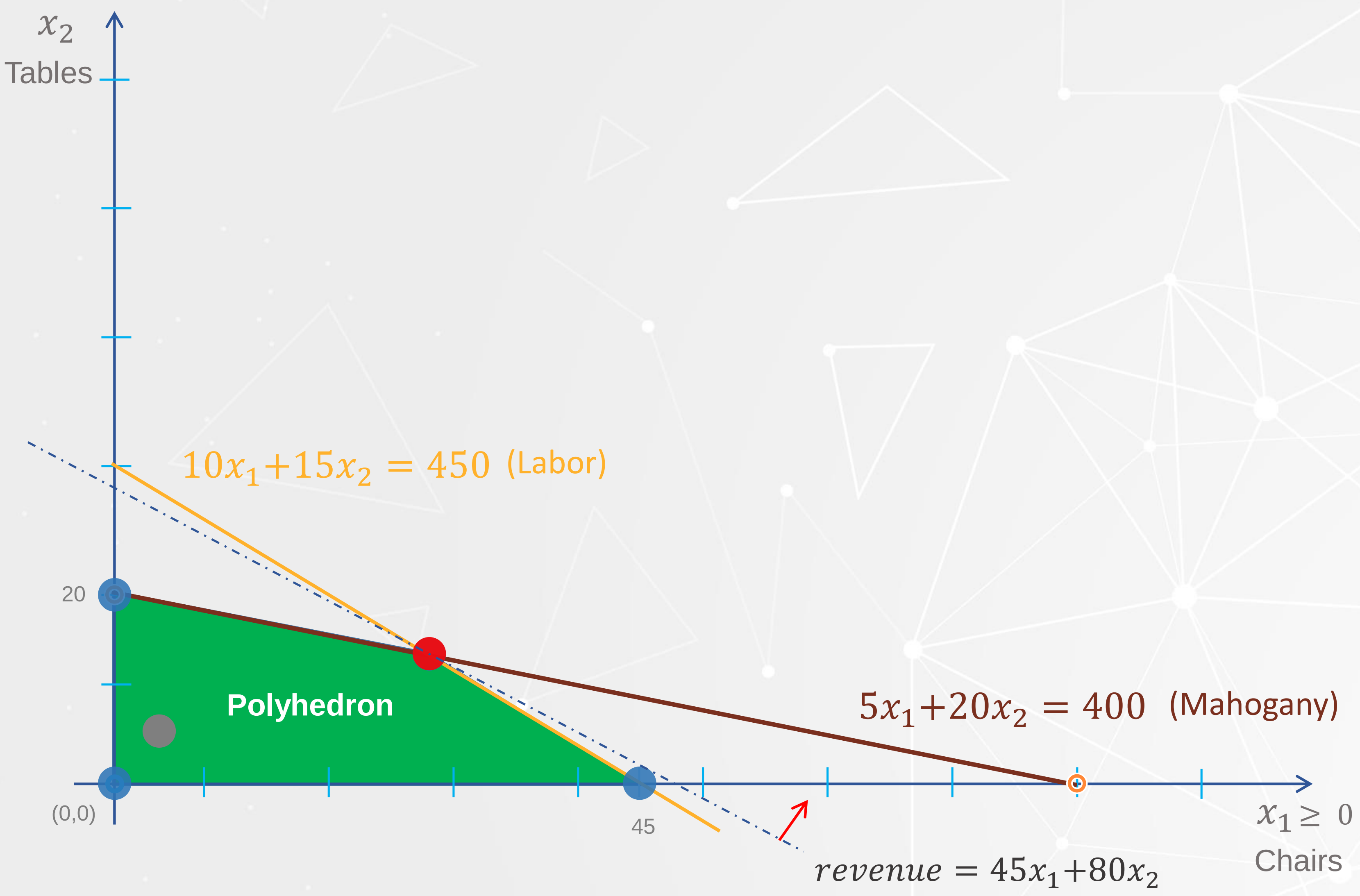
Efficient Production Plan

Mahogany slack variable
h1 = 400 – 5(x1=24) – 20(x2=14)= 0

Labor slack variable
h2 = 450 – 10(x1=24) – 15(x2=14) = 0

Break

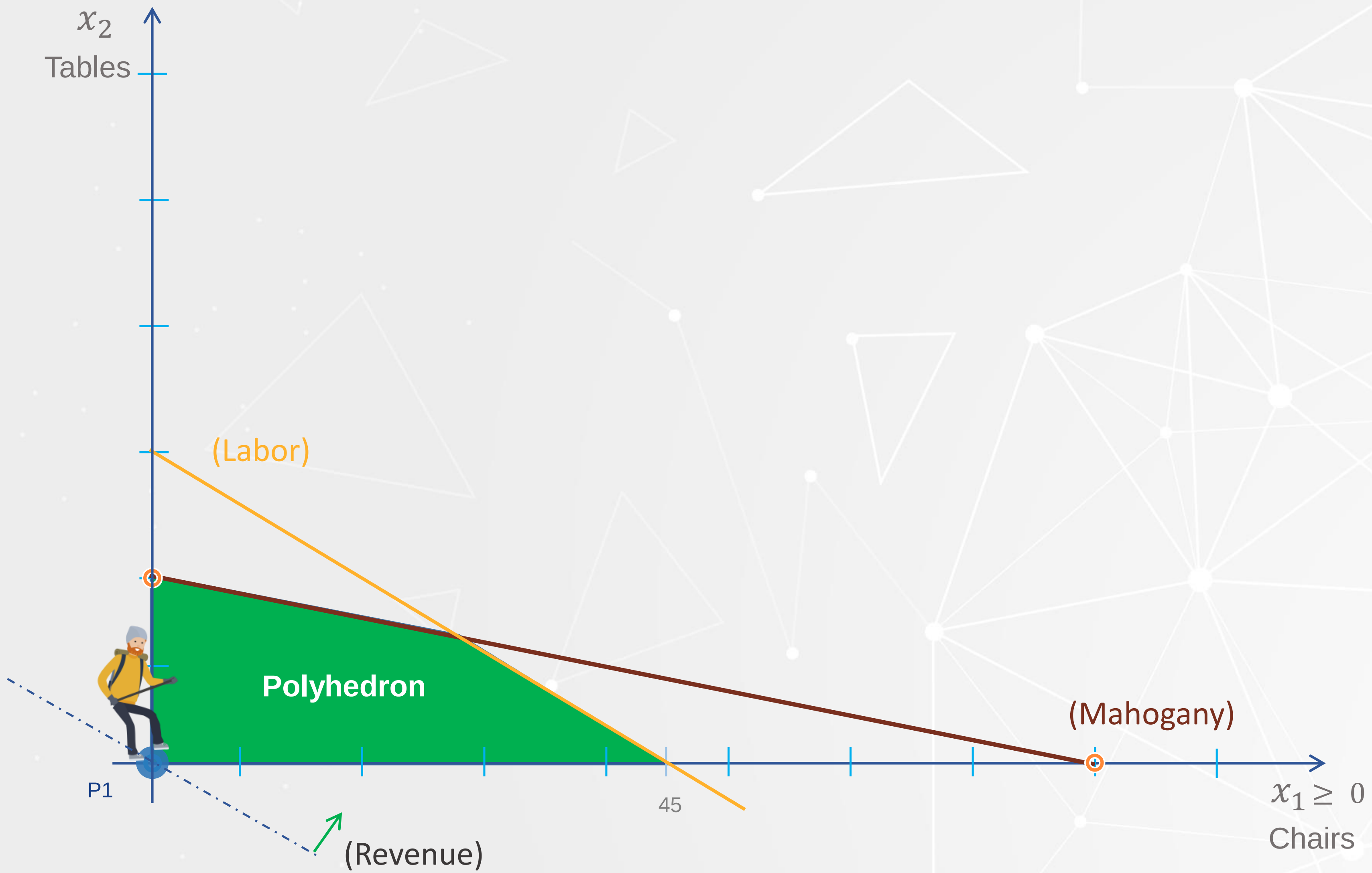
Fundamental theorem of linear programming ..



- **Definitions**:
  - A **solution** of an LP problem is a set of values of the decision variables that satisfies all the constraints of the problem defined by the polyhedron.
  - A **corner point solution** is a vertex of the polyhedron defining the feasible region of the LP problem.
  - An **optimal solution** is a solution of the LP problem that cannot be improved.
- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.

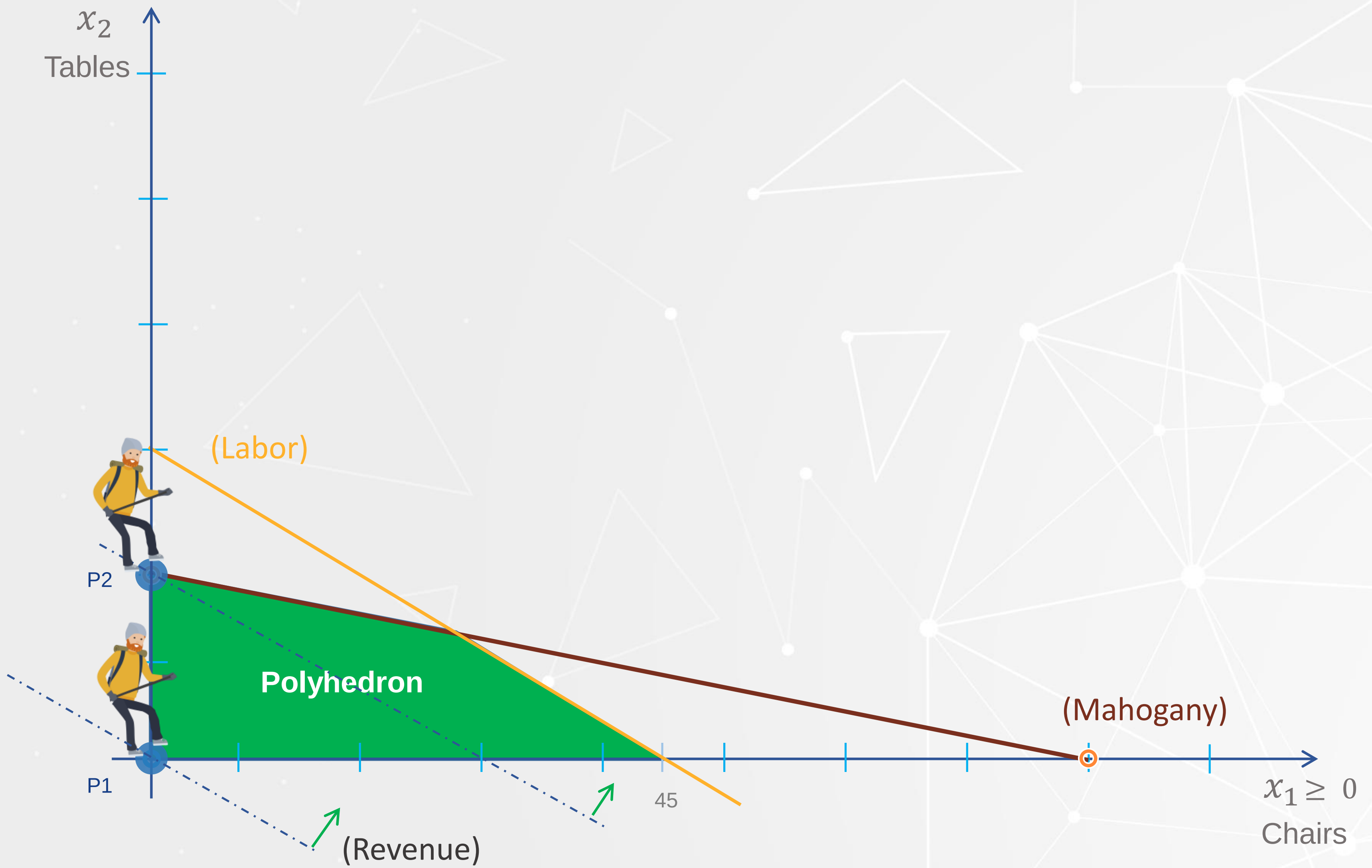
GUROBI OPTIMIZATION

Fundamental theorem of linear programming .. 2



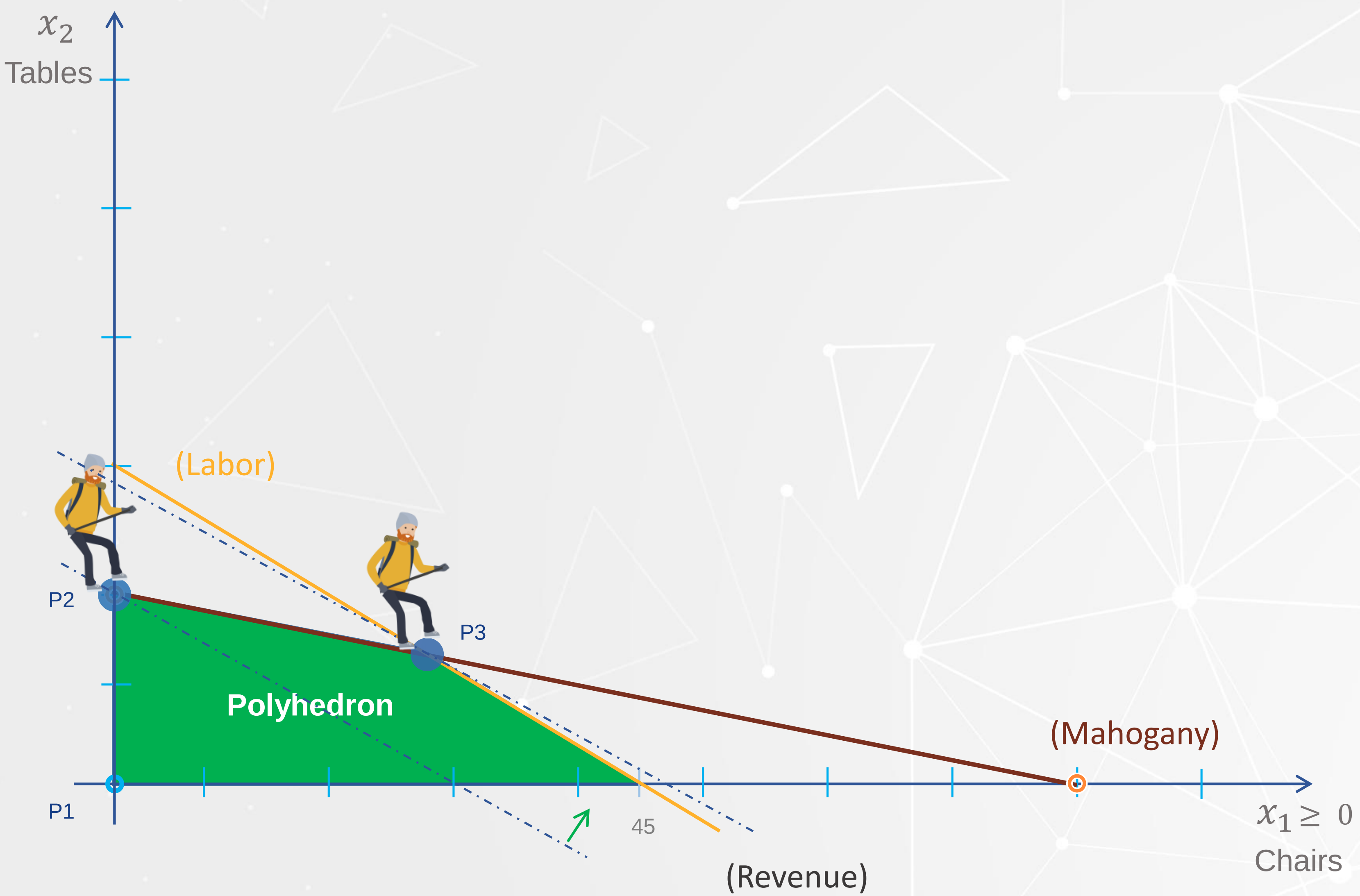
- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.
- Initial corner point solution P1 =(0 chairs, 0 tables)

Fundamental theorem of linear programming .. 2



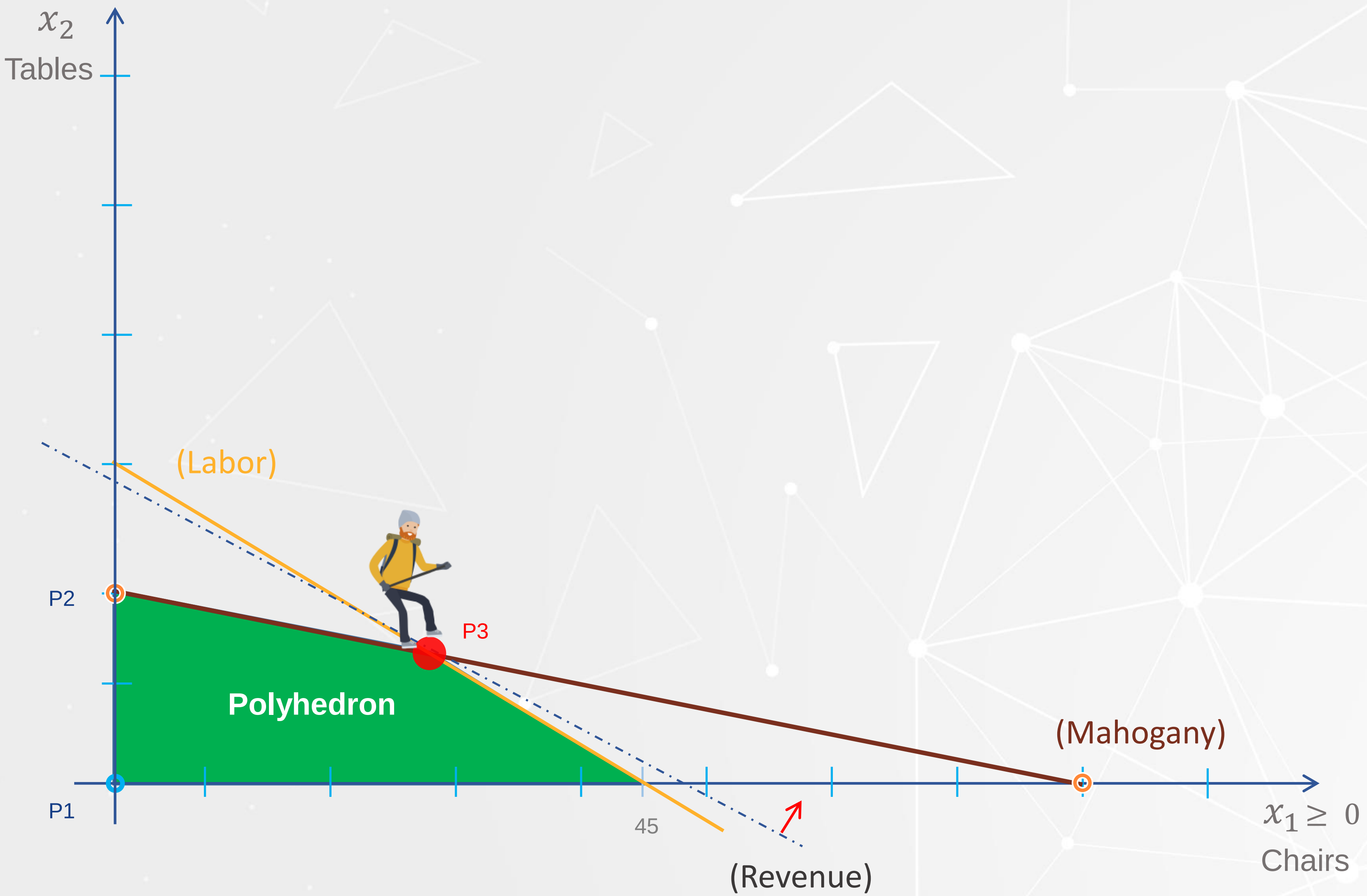
- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.
- Initial corner point solution
  P1 =(0 chairs, 0 tables)
- Adjacent corner point solution
  P2 =(0 chairs, 20 tables)

Fundamental theorem of linear programming .. 2



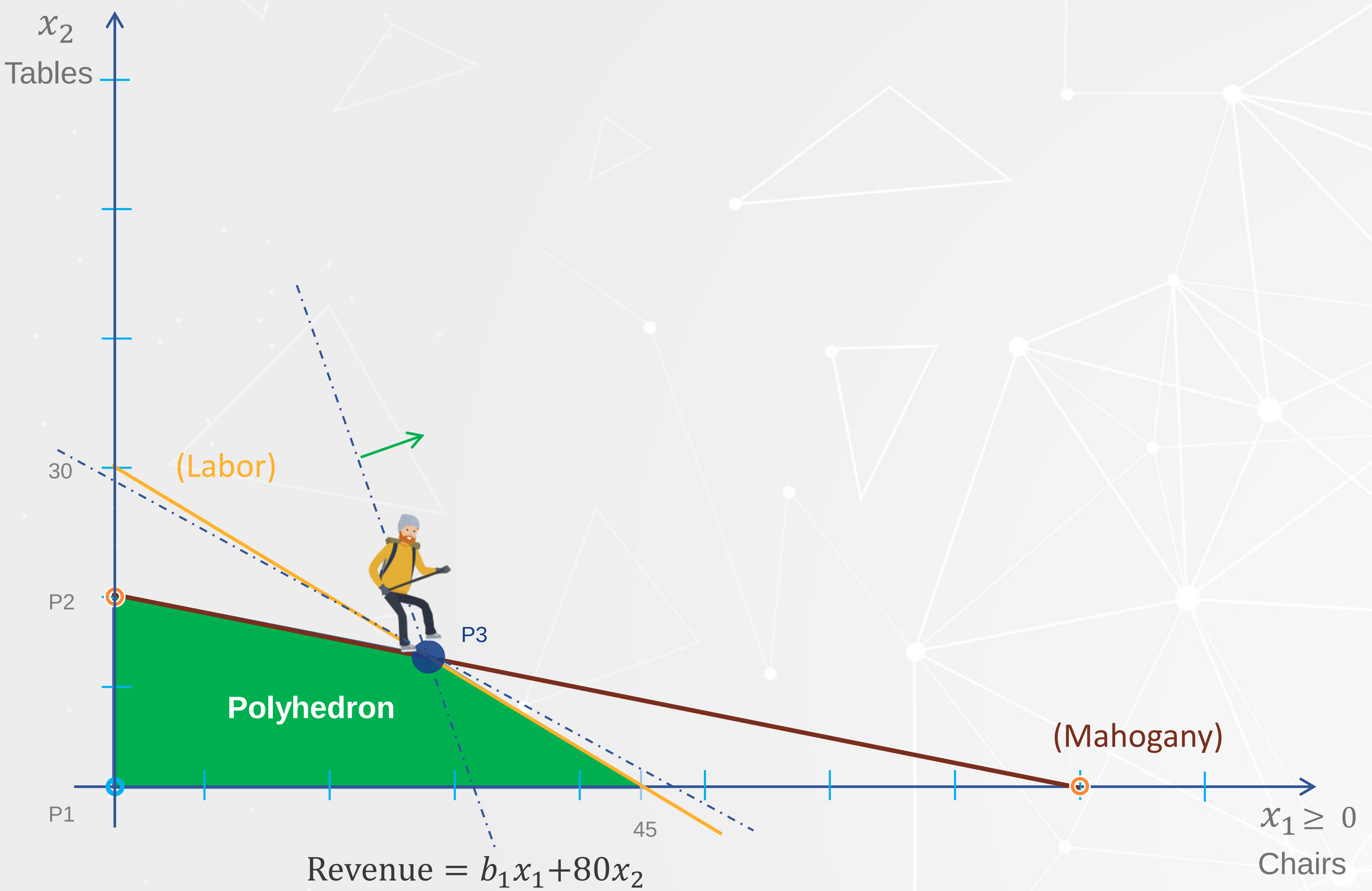
- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.
- Initial corner point solution
  P1 =(0 chairs, 0 tables)
- Adjacent corner point solution
  P2 =(0 chairs, 20 tables)
- Adjacent corner point solution
  P3 =(24 chairs, 14 tables)

Fundamental theorem of linear programming .. 2



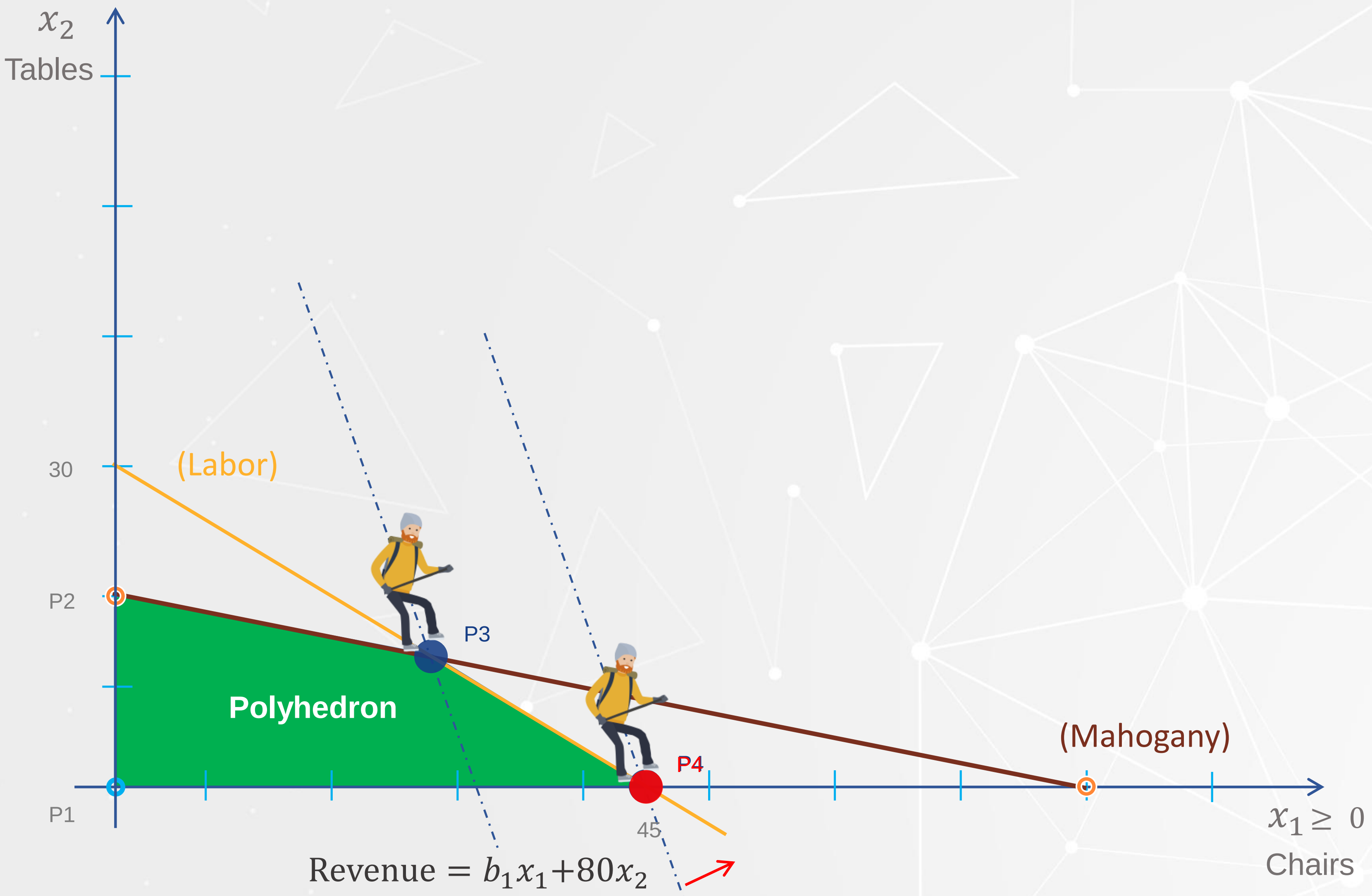
- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.
- Initial corner point solution
  P1 =(0 chairs, 0 tables)
- Adjacent corner point solution
  P2 =(0 chairs, 20 tables)
- Adjacent corner point solution
  P3 =(24 chairs, 14 tables)

Fundamental theorem of linear programming .. 3



- Significant increase in the price (b1) of chairs.
- New oportunity to increase revenue

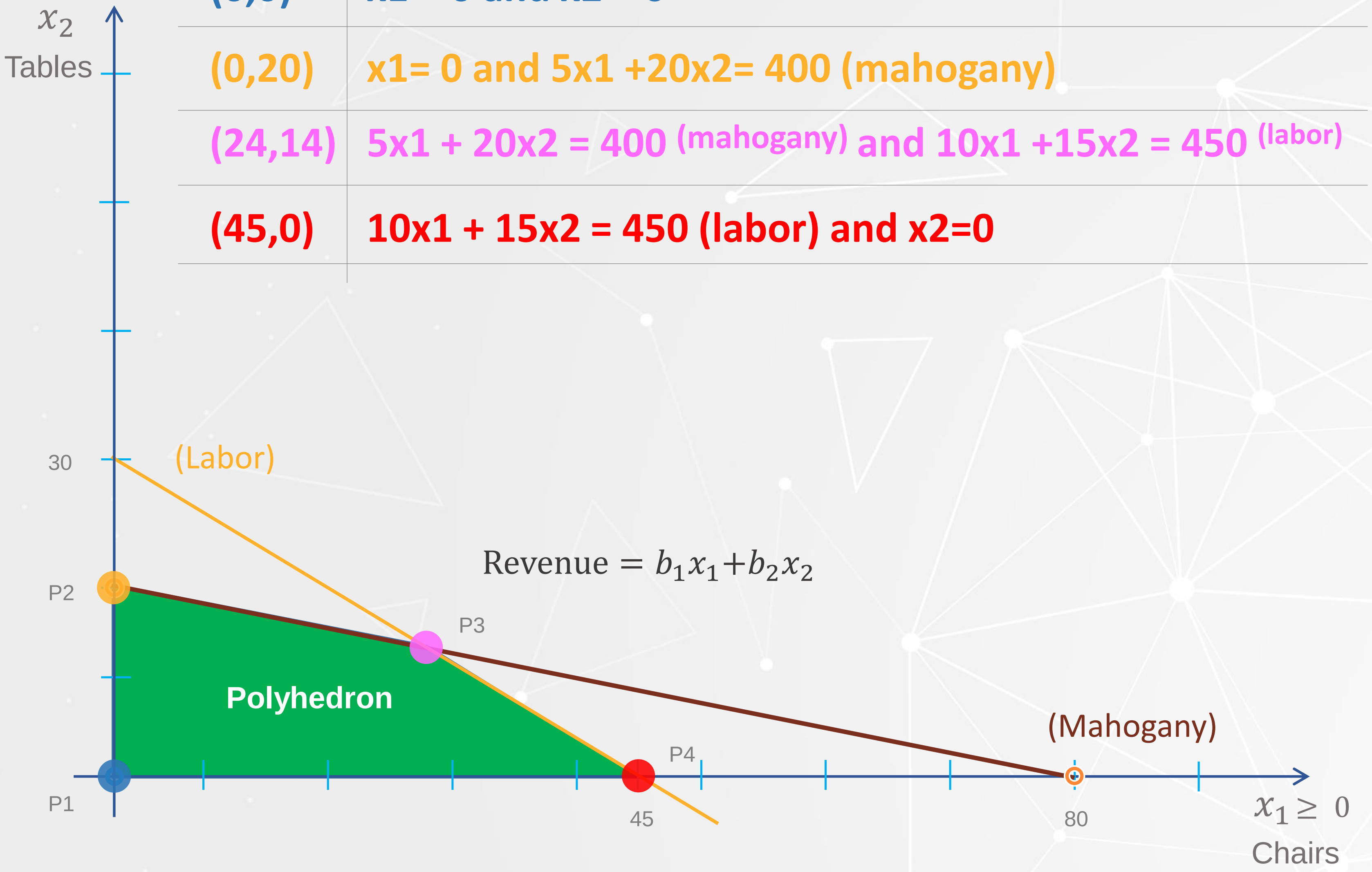
Fundamental theorem of linear programming .. 3



- The new Production Plan P4 = (45 chairs, 0 tables) is optimal

Fundamental theorem of linear programming .. 4

Vertex	Equations
(0,0)	x1 = 0 and x2 = 0
(0,20)	x1= 0 and 5x1 +20x2= 400 (mahogany)
(24,14)	5x1 + 20x2 = 400 (mahogany) and 10x1 +15x2 = 450 (labor)
(45,0)	10x1 + 15x2 = 450 (labor) and x2=0



- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.

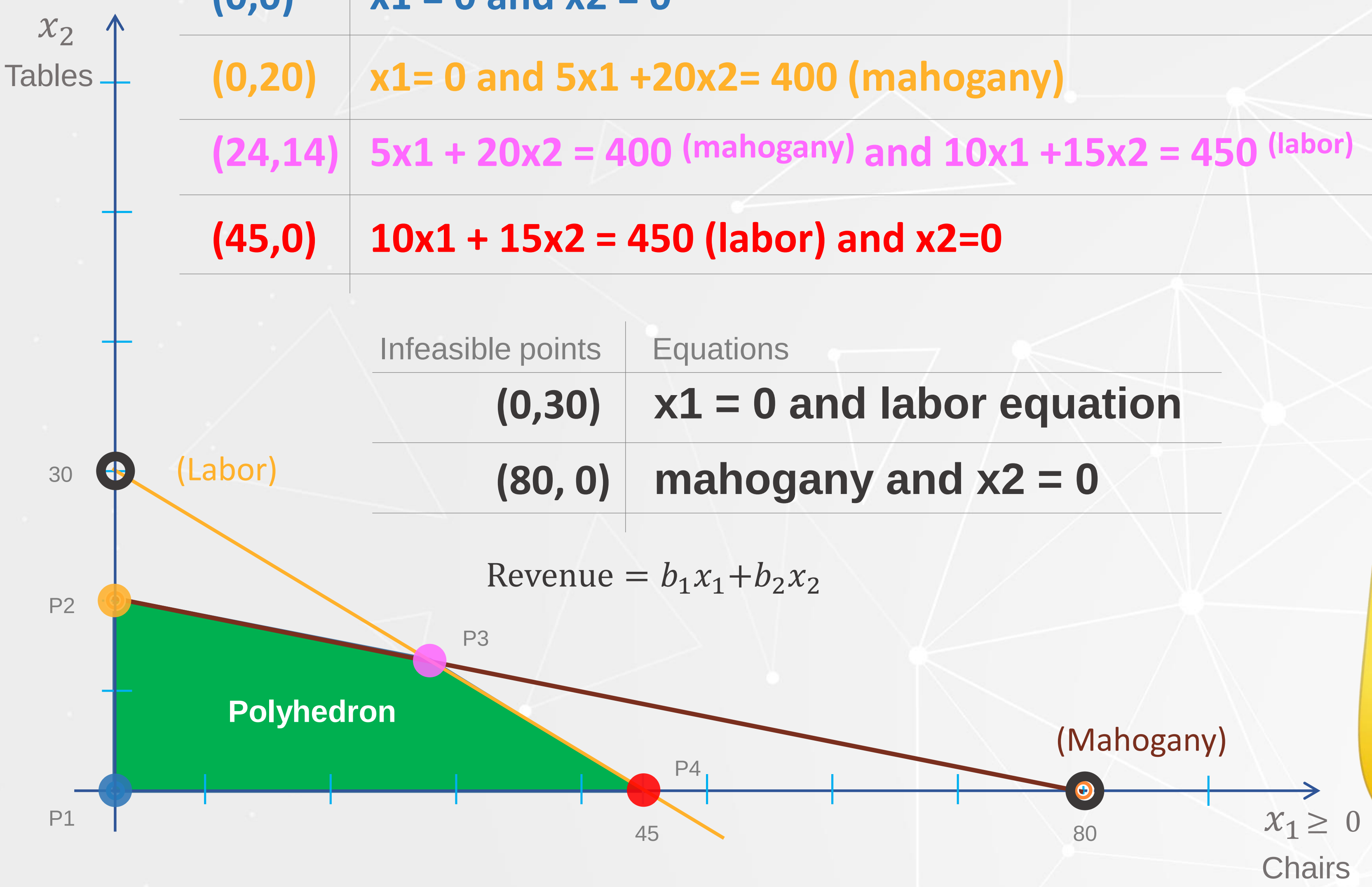
Note that the vertices (corner points) of the polyhedron are the solution of a system of equations.

GUROBI OPTIMIZATION

Fundamental theorem of linear programming .. 5

Vertex	Equations
(0,0)	x1 = 0 and x2 = 0
(0,20)	x1= 0 and 5x1 +20x2= 400 (mahogany)
(24,14)	5x1 + 20x2 = 400 (mahogany) and 10x1 +15x2 = 450 (labor)
(45,0)	10x1 + 15x2 = 450 (labor) and x2=0

Infeasible points	Equations
(0,30)	x1 = 0 and labor equation
(80, 0)	mahogany and x2 = 0



$$\text{Revenue} = b_1 x_1 + b_2 x_2$$

- **Theorem**:
  - If a linear programming problem has an optimal solution, there is at least one optimal solution that is a corner point solution.

Note that the vertices (corner points) of the polyhedron are the solution of a system of equations.

Also, observe that there are other points that are the solutions of a system of equations, although these points are infeasible because they are not vertices of the polyhedron.

Enumeration approach

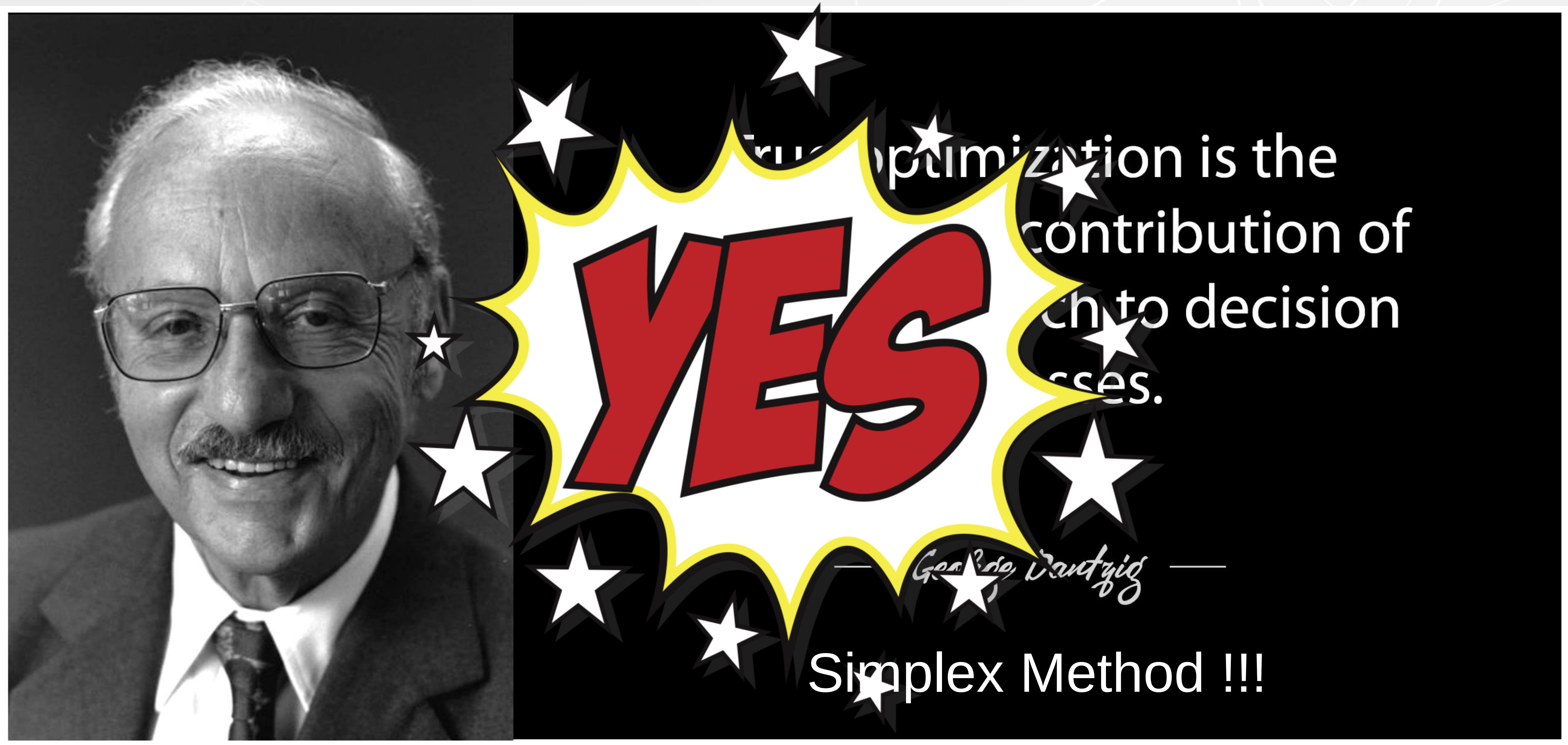
Enumeration of solutions of the system of equations for the furniture problem

Points of interest	Vertex of the polyhedron	Objective function value. Revenue = 45x1 + 80x2
(0,0)	Yes (feasible)	0 = 45*0 + 80*0
(0,20)	Yes (feasible)	1600 = 45*0 + 80*20
(24,14)	Yes (feasible)	2200 = 45*24 + 80*14 Optimal!!
(45,0)	Yes (feasible)	2025 = 45*45 + 80*0
(0,30)	No (infeasible)	2400 = 45*30 + 80*30
(80,0)	No (infeasible)	3600 = 45*80 + 80*0

$\sim 10^{88}$

This number is larger than the number of atoms ($\sim 10^{80}$) in the known universe !!

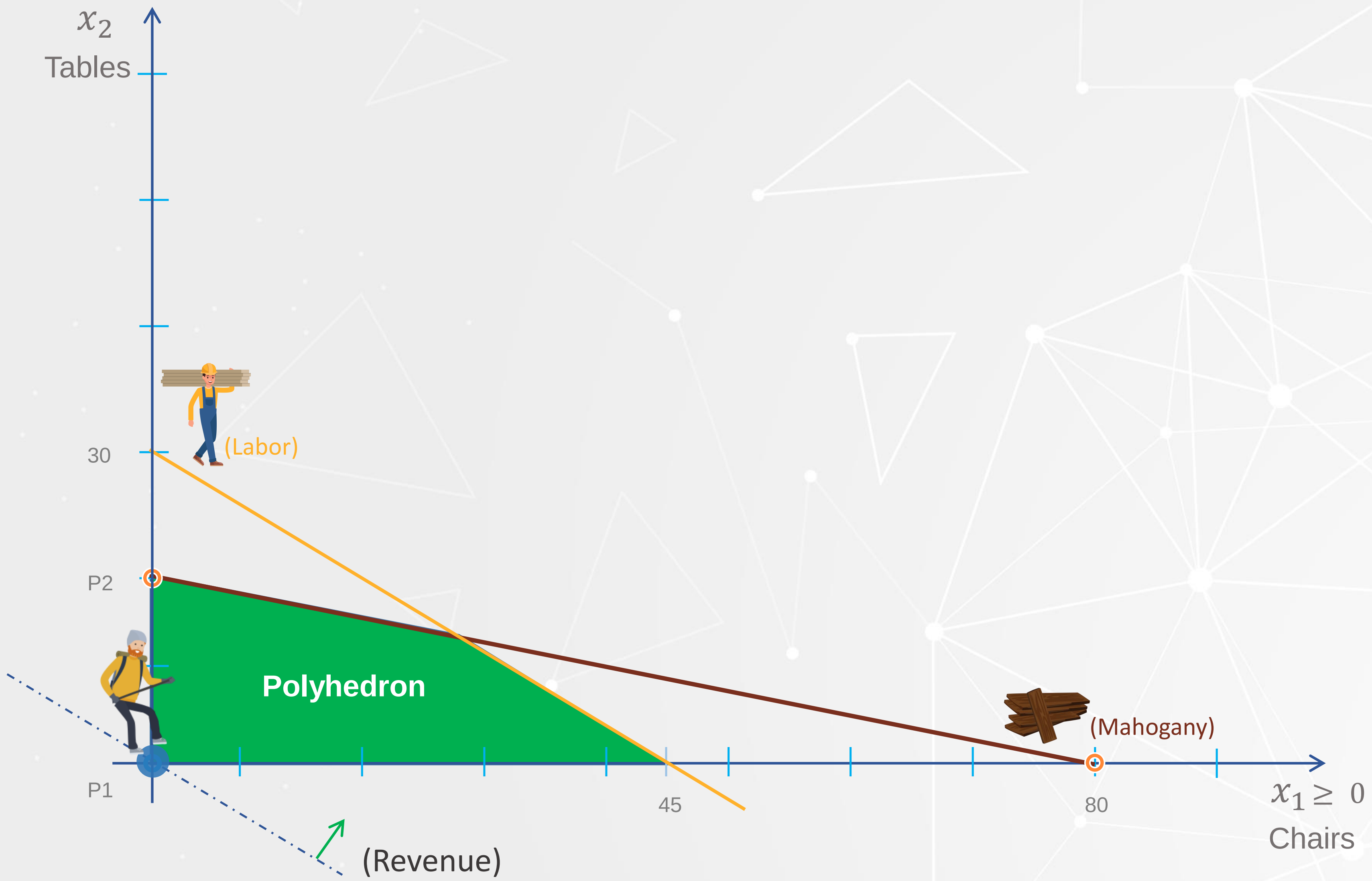
Is there a way that we can traverse vertices in the polyhedron in a more efficient way?



Simplex method

Overview

Linear Programming



Furniture problem LP problem formulation

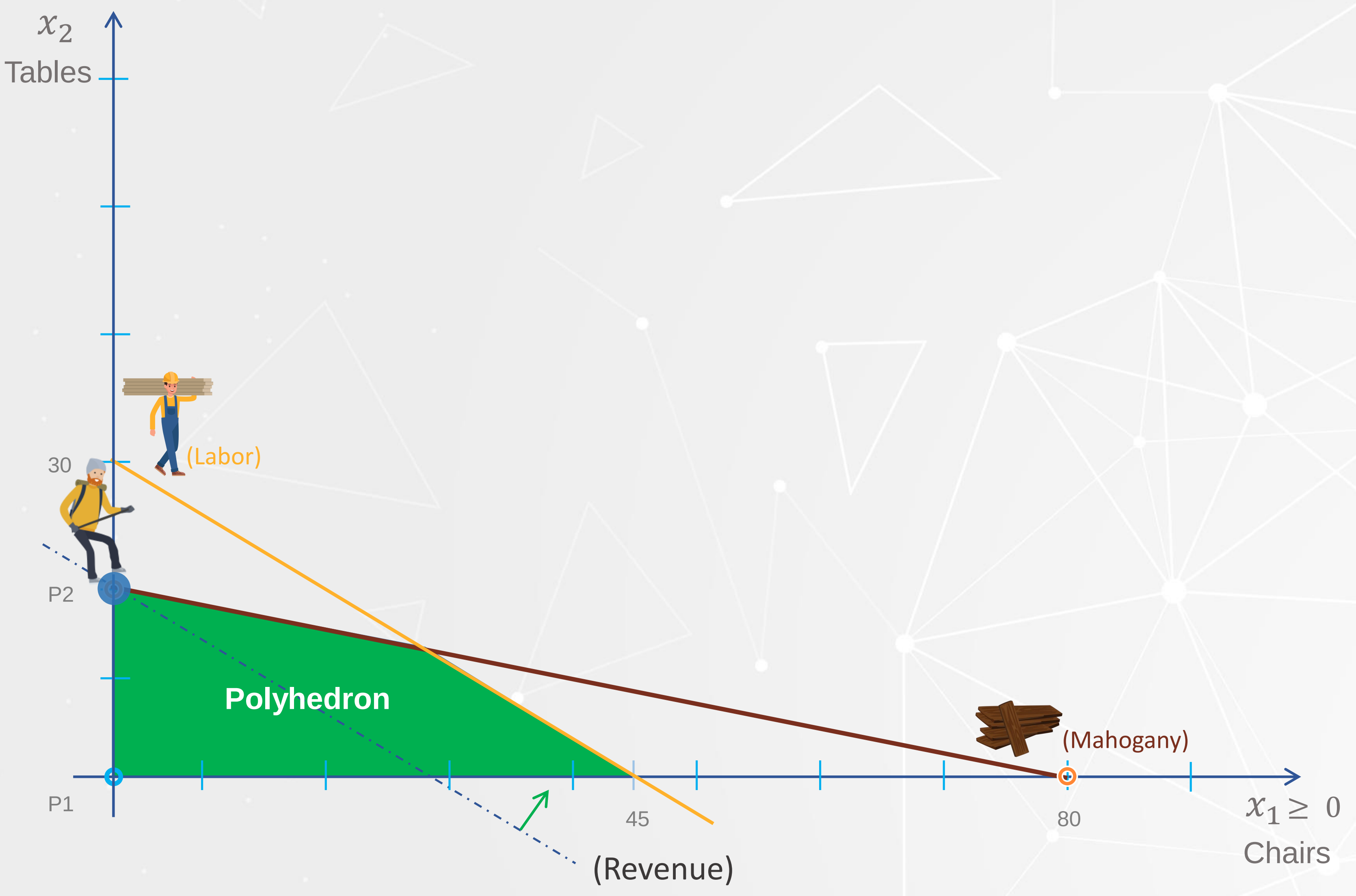
$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Mahogany}$$

$$(3.0).\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor}$$

$$\boldsymbol{x_1, x_2 \geq 0} \quad \text{Non − negativity}$$

Linear Programming



Furniture problem LP problem formulation

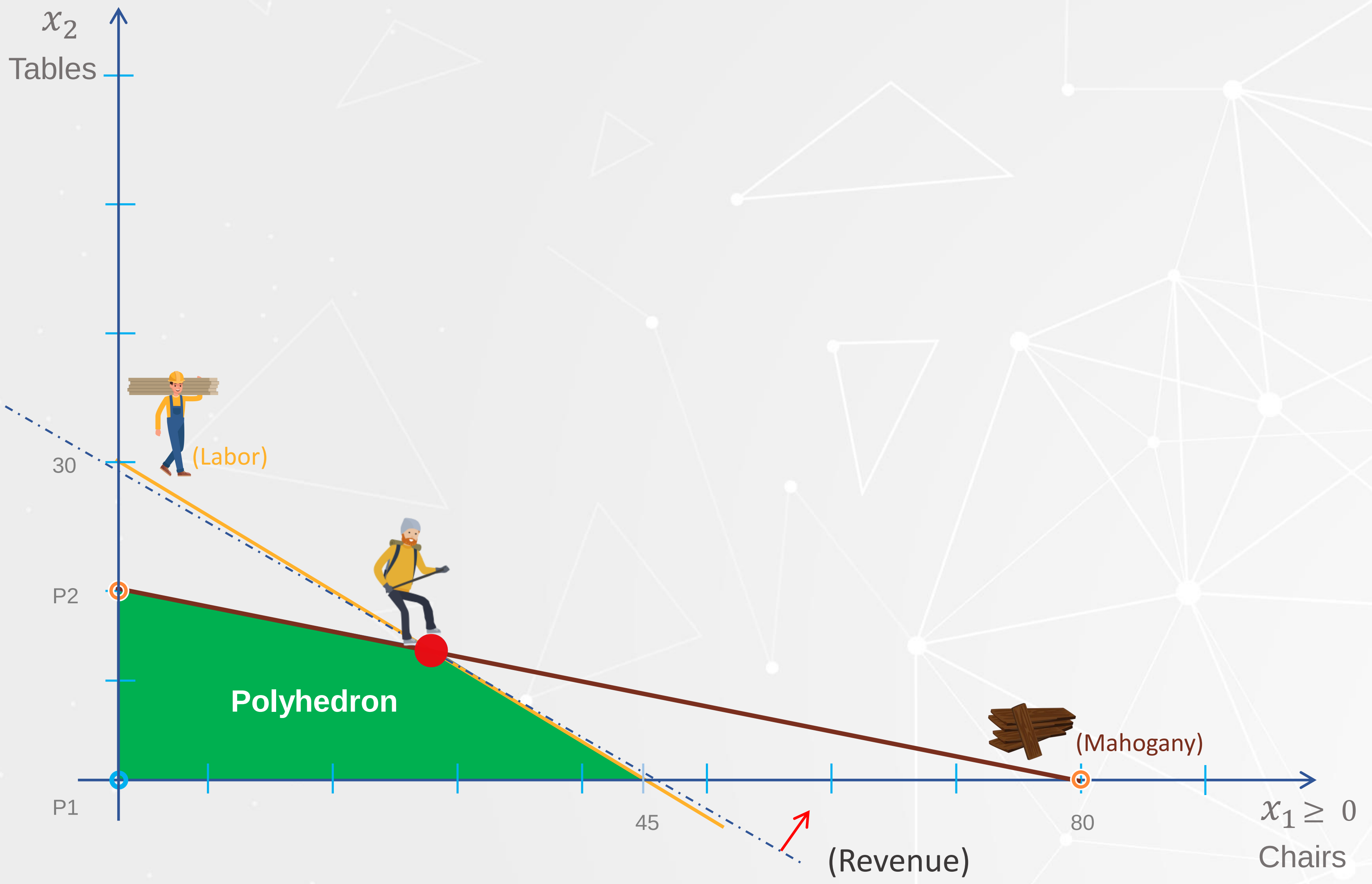
(1.0). Max revenue $= 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Mahogany

(3.0). $10x_1 + 15x_2 \leq 450$ Labor

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

Linear Programming



Furniture problem LP problem formulation

(1.0). $\text{Max revenue} = 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 \leq 400$ Mahogany

(3.0). $10x_1 + 15x_2 \leq 450$ Labor

$\boldsymbol{x_1, x_2 \geq 0}$ Non – negativity

Linear Programming/Simplex Method

We call the formulation of an LP problem the **original LP problem**

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \qquad \text{Mahogany}$$

$$(3.0).\quad 10x_1 + 15x_2 \leq 450 \qquad \text{Labor}$$

$$\boldsymbol{x_1, x_2 \geq 0} \qquad \text{Non} - \text{negativity}$$

Linear Programming/Simplex Method

The original LP problem in a **standard form** is:

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 + h_1 = 400 \quad \text{mahogany}$$

$$(3.0).\quad 10x_1 + 15x_2 + h_2 = 450 \quad \text{Labor}$$

$$\boldsymbol{x_1, x_2, h_1, h_2 \geq 0} \quad \text{Non} - \text{negativity}$$

Original LP problem.

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{mahogany}$$

$$(3.0).\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor}$$

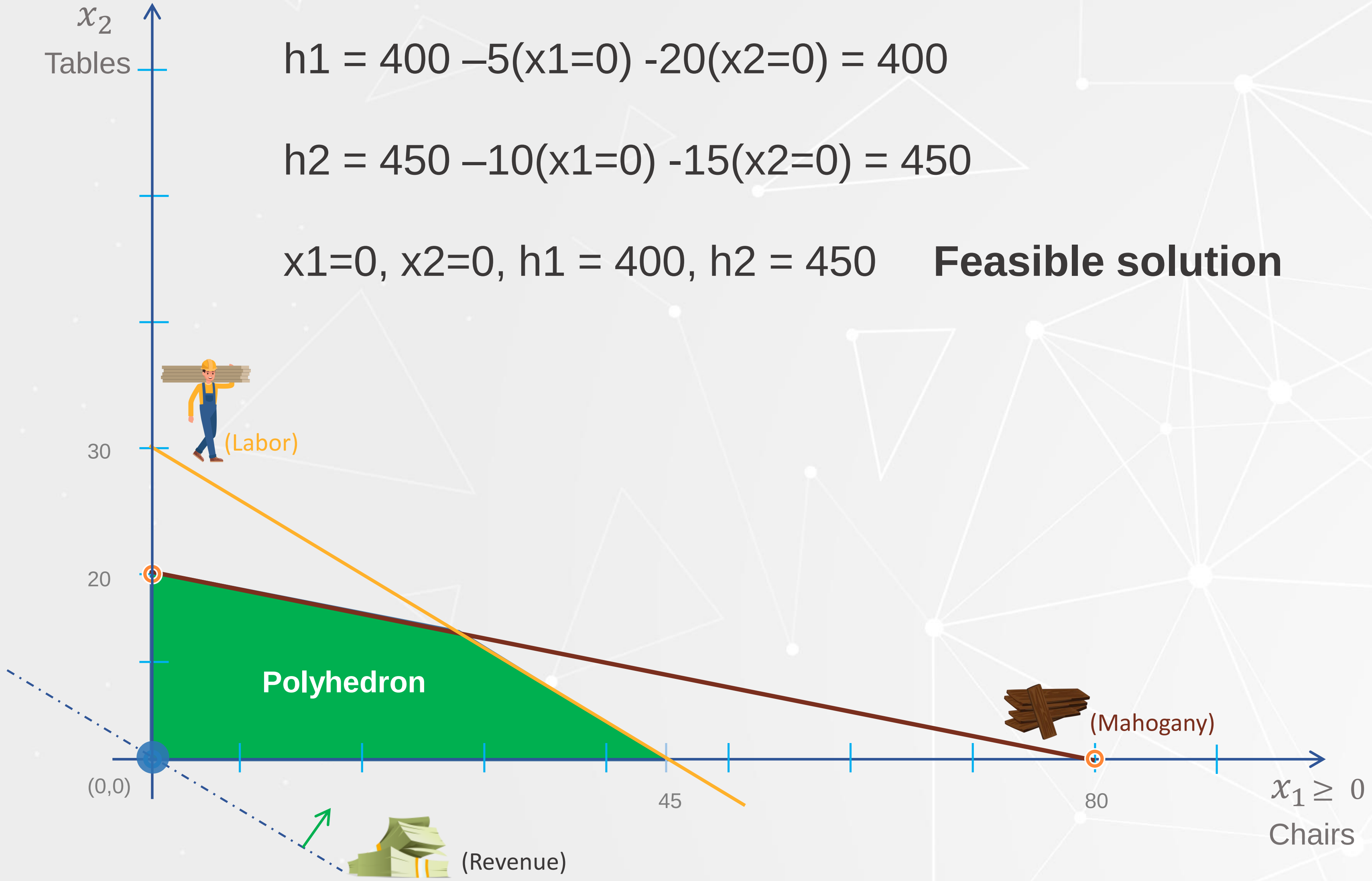
$$\boldsymbol{x_1, x_2 \geq 0} \quad \text{Non} - \text{negativity}$$

Linear Programming/Simplex Method .. 2

h1 = 400 –5(x1=0) -20(x2=0) = 400

h2 = 450 –10(x1=0) -15(x2=0) = 450

x1=0, x2=0, h1 = 400, h2 = 450 **Feasible solution**



Furniture **problem standard form**

(1.0). $\text{Max revenue} = 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 + h_1 = 400$ Mahogany

(3.0). $10x_1 + 15x_2 + h_2 = 450$ Labor

$\boldsymbol{x_1, x_2, h_1, h_2 \geq 0}$ Non – negativity

(x1=0, x2=0) initial solution
Revenue = 0

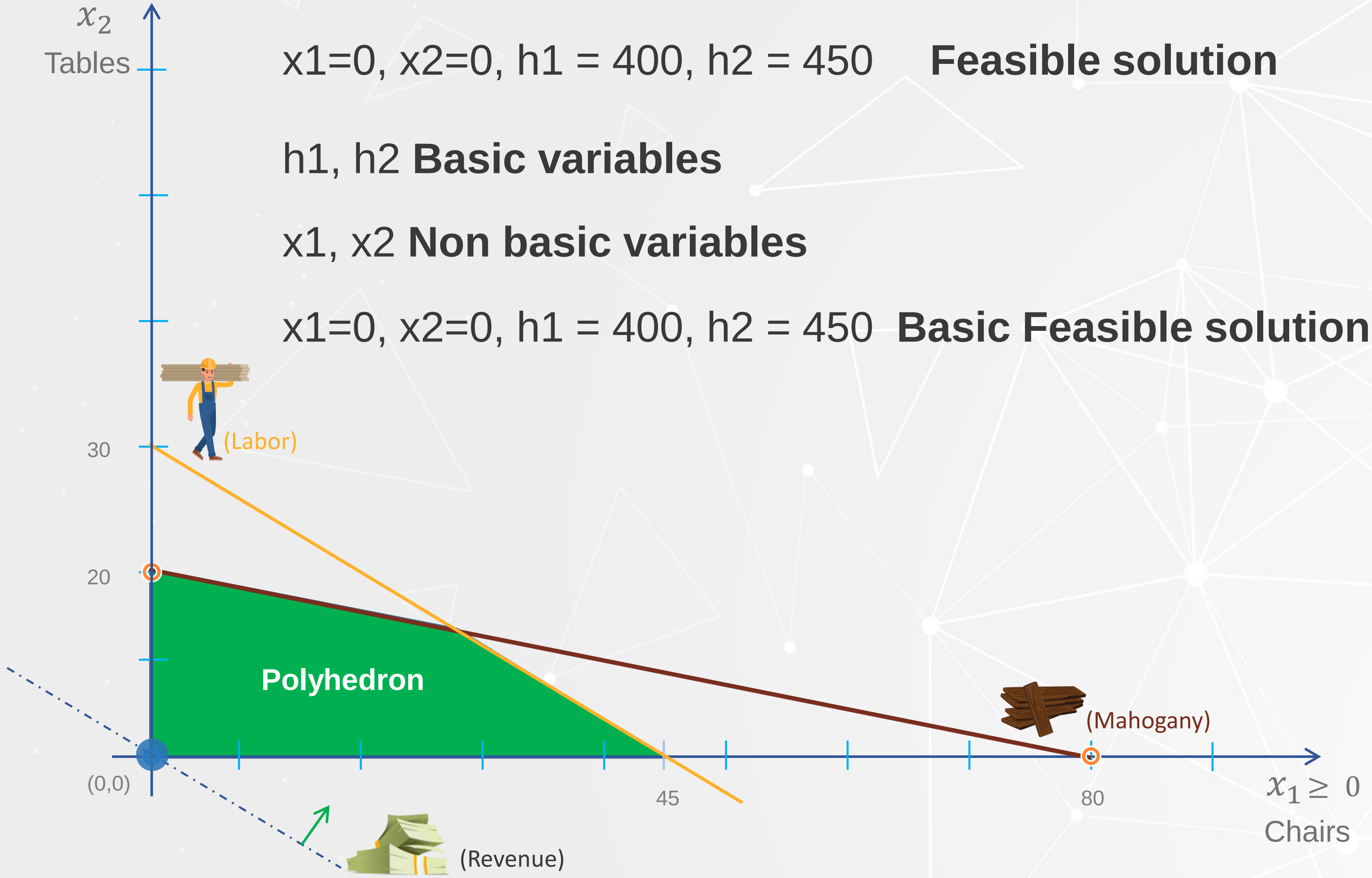
Linear Programming/Simplex Method .. 2

x1=0, x2=0, h1 = 400, h2 = 450 **Feasible solution**

h1, h2 **Basic variables**

x1, x2 **Non basic variables**

x1=0, x2=0, h1 = 400, h2 = 450 **Basic Feasible solution**



Furniture **problem standard form**

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 + h_1 = 400 \quad \text{Mahogany}$$

$$(3.0).\quad 10x_1 + 15x_2 + h_2 = 450 \quad \text{Labor}$$

$$\boldsymbol{x_1, x_2, h_1, h_2 \geq 0} \quad \text{Non – negativity}$$

Linear Programming/Simplex Method .. 3

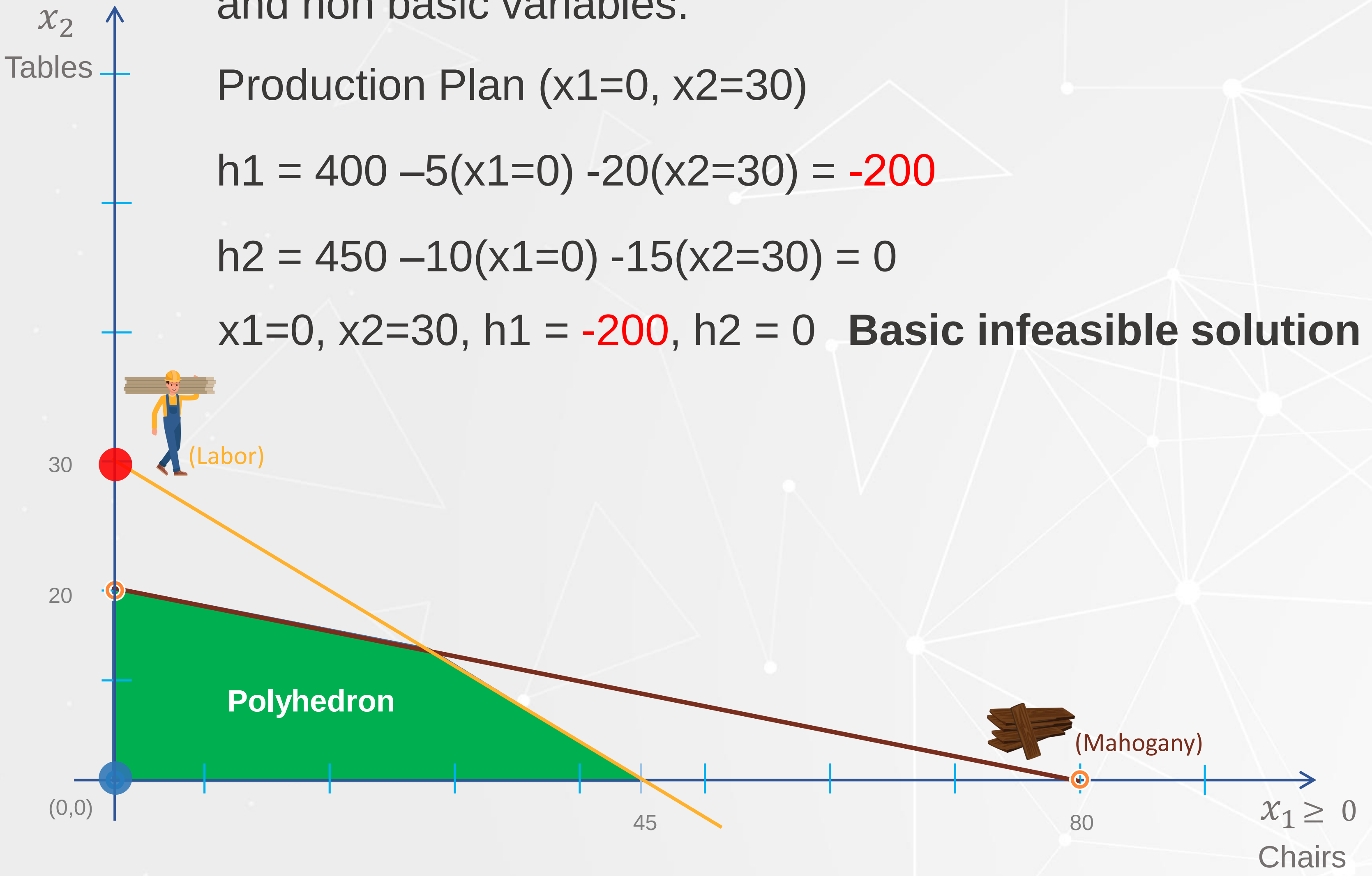
A basic solution is defined by the values of the basic and non basic variables.

Production Plan (x1=0, x2=30)

h1 = 400 –5(x1=0) -20(x2=30) = -200

h2 = 450 –10(x1=0) -15(x2=30) = 0

x1=0, x2=30, h1 = -200, h2 = 0 **Basic infeasible solution**



Furniture **problem standard form**

(1.0). $\text{Max revenue} = 45x_1 + 80x_2$

(2.0) $5x_1 + 20x_2 + h_1 = 400$ Mahogany

(3.0). $10x_1 + 15x_2 + h_2 = 450$ Labor

$\boldsymbol{x_1, x_2, h_1, h_2 \geq 0}$ Non – negativity

Linear Programming/Simplex Method .. 4

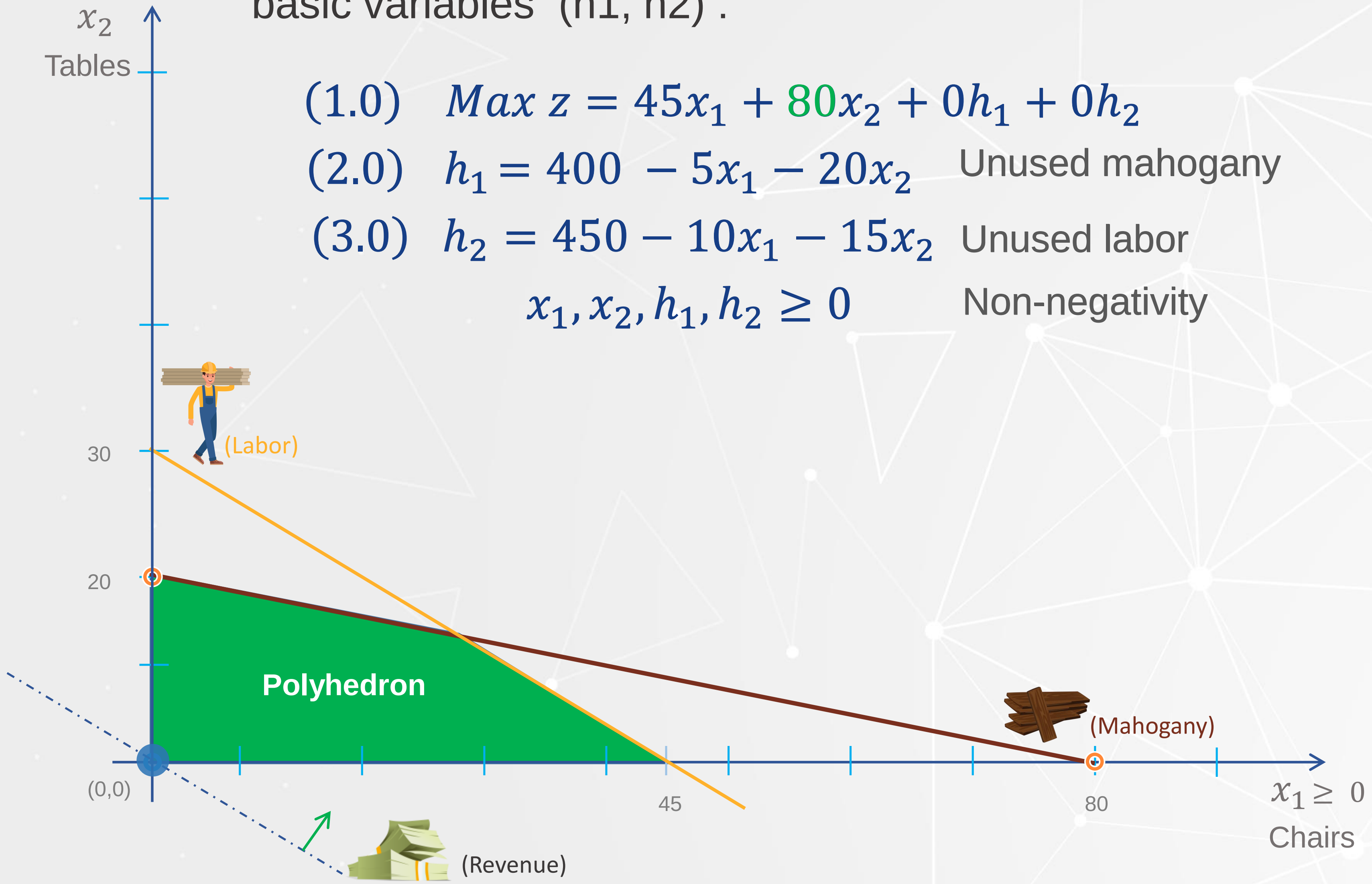
LP problem in a **canonical form** with respect to the basic variables (h1, h2) :

$$(1.0)\quad Max\ z = 45x_1 + 80x_2 + 0h_1 + 0h_2$$

$$(2.0)\quad h_1 = 400 - 5x_1 - 20x_2 \quad \text{Unused mahogany}$$

$$(3.0)\quad h_2 = 450 - 10x_1 - 15x_2 \quad \text{Unused labor}$$

$$x_1, x_2, h_1, h_2 \geq 0 \quad \text{Non-negativity}$$



Furniture **problem standard form**

$$(1.0).\quad \text{Max revenue} = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 + h_1 = 400 \quad \text{Mahogany}$$

$$(3.0).\quad 10x_1 + 15x_2 + h_2 = 450 \quad \text{Labor}$$

$$\boldsymbol{x_1, x_2, h_1, h_2 \geq 0} \quad \text{Non - negativity}$$

Reduced costs:
Objective funcion coefficients of non basic variables (x1, x2)

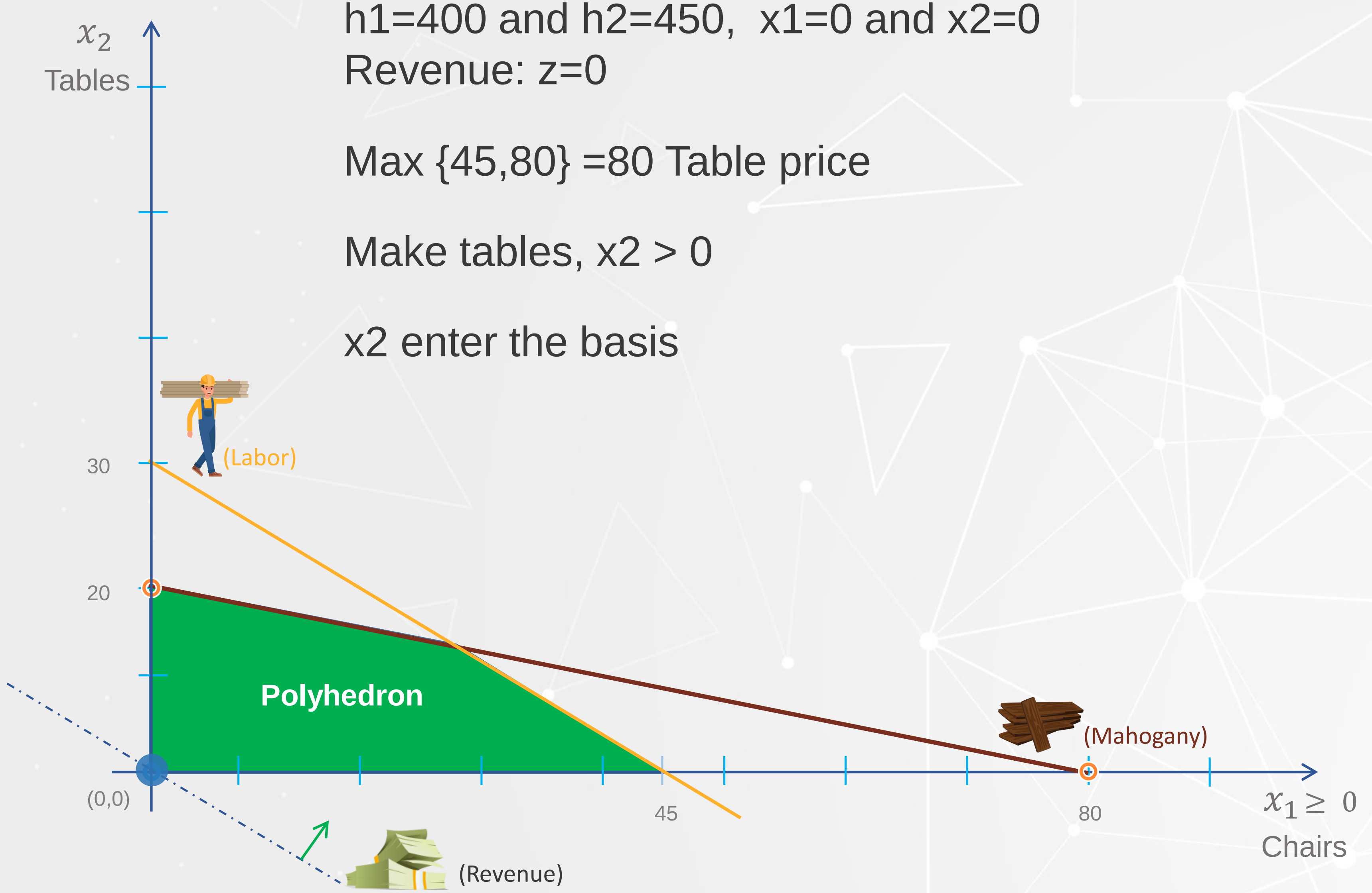
Linear Programming/Simplex Method .. 5

Current basic feasible solution:
h1=400 and h2=450, x1=0 and x2=0
Revenue: z=0

Max {45,80} =80 Table price

Make tables, x2 > 0

x2 enter the basis



Furniture problem **canonical form**

$$(1.0)\quad Max\ z = 45x_1 + 80x_2 + 0h_1 + 0h_2$$

$$(2.0)\quad h_1 = 400 - 5x_1 - 20x_2 \quad \text{Mahogany}$$

$$(3.0)\quad h_2 = 450 - 10x_1 - 15x_2 \quad \text{Labor}$$

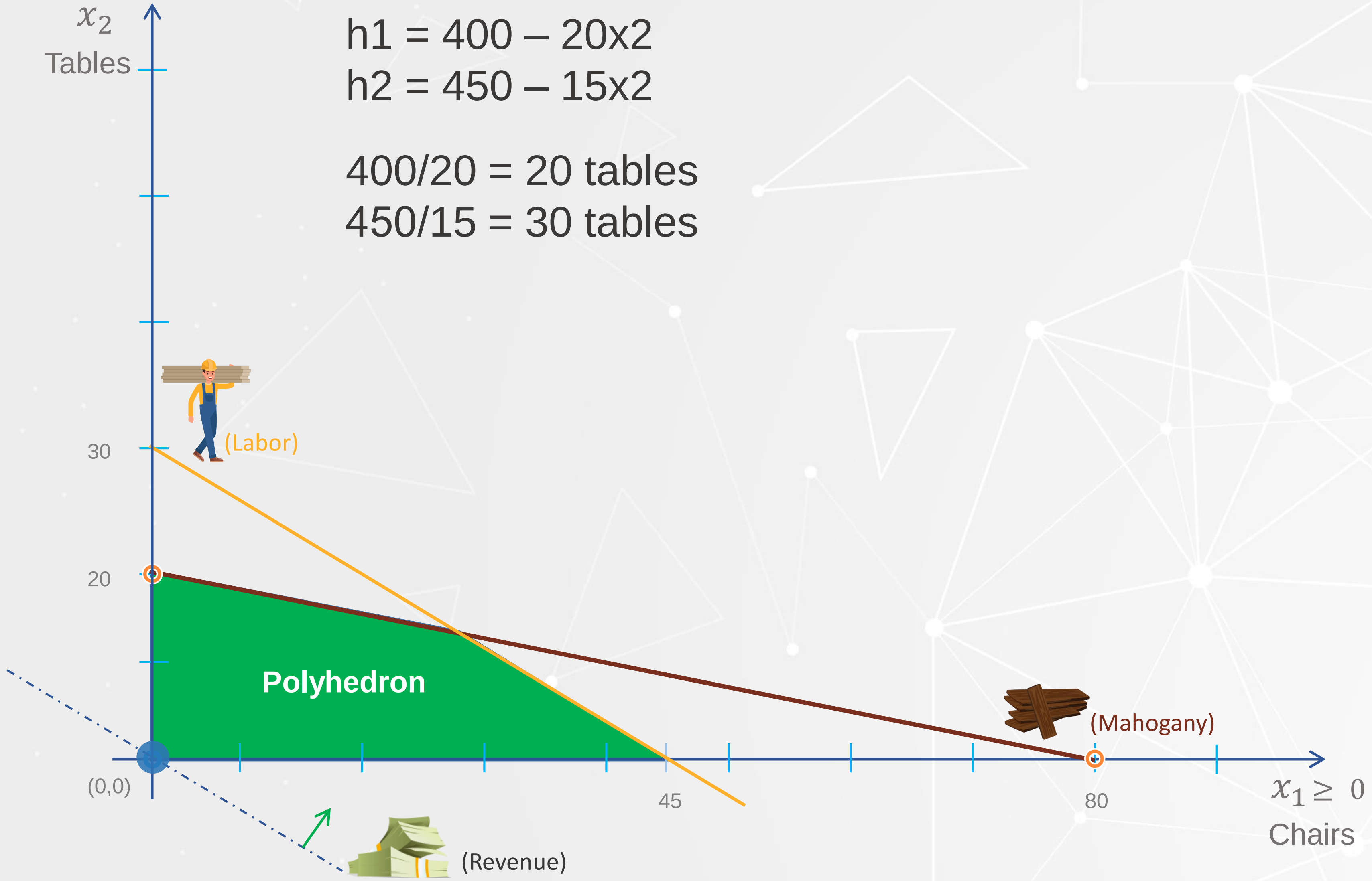
$$x_1, x_2, h_1, h_2 \geq 0 \quad \text{Non-negativity}$$

Linear Programming/Simplex Method .. 5

How many tables (x2) can we make?

h1 = 400 – 20x2
h2 = 450 – 15x2

400/20 = 20 tables
450/15 = 30 tables



Furniture problem **canonical form**

$$(1.0)\quad Max\ z = 45x_1 + 80x_2 + 0h_1 + 0h_2$$

$$(2.0)\quad h_1 = 400 - 5x_1 - 20x_2$$ Mahogany

$$(3.0)\quad h_2 = 450 - 10x_1 - 15x_2$$ Labor

$$x_1, x_2, h_1, h_2 \geq 0$$ Non-negativity

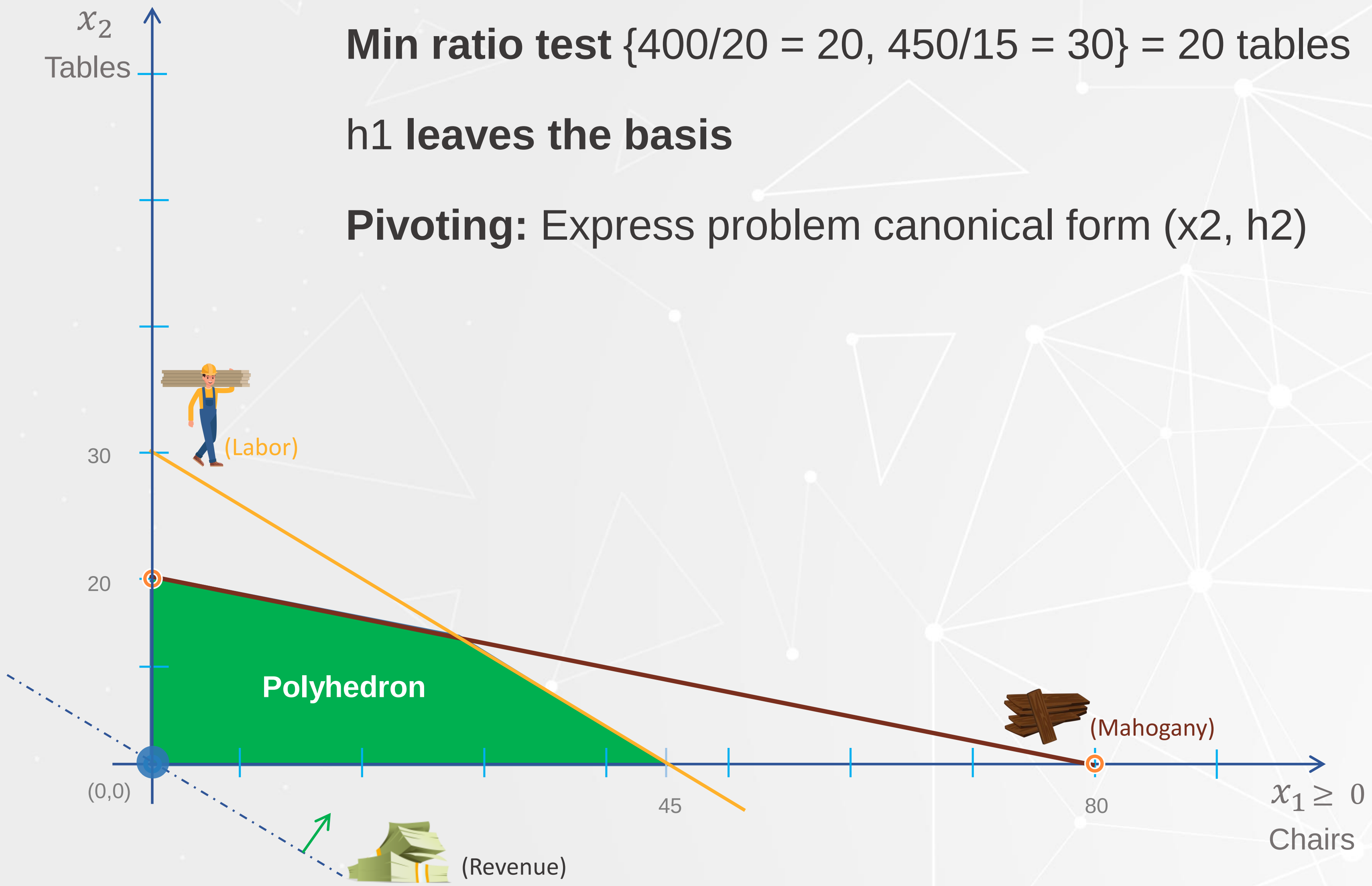
Linear Programming/Simplex Method .. 5

If x2 =30, h1 = 400 – 20(x2=30) = -200!!!!

Min ratio test {400/20 = 20, 450/15 = 30} = 20 tables

h1 **leaves the basis**

Pivoting: Express problem canonical form (x2, h2)



Furniture problem **canonical form**

(1.0) $Max\ z = 45x_1 + 80x_2 + 0h_1 + 0h_2$

(2.0) $h_1 = 400 - 5x_1 - 20x_2$ Mahogany

(3.0) $h_2 = 450 - 10x_1 - 15x_2$ Labor

$x_1, x_2, h_1, h_2 \geq 0$ Non-negativity

Linear Programming/Simplex Method (Pivoting) .. 6

$$(1.0) \quad Max\ z = 45x_1 + 80x_2 + 0h_1 + 0h_2$$

$$(2.0) \quad h_1 = 400 - 5x_1 - 20x_2$$

$$(3.0) \quad h_2 = 450 - 10x_1 - 15x_2$$

$$x_1, x_2, h_1, h_2 \geq 0$$

In equation 2.0, express x2 in terms of x1 and h1

$$(2.0) \quad x_2 = 20 - \left({}^{1}/_{4}\right)x_1 - \left({}^{1}/_{20}\right)h_1$$

Linear Programming/Simplex Method (Pivoting) .. 6

In equation 2.0, express x2 in terms of x1 and h1

$$(2.0) \quad x_2 = 20 - ({}^1/_4)x_1 - ({}^1/_{20})h_1$$

We substitute the value of x2 in equation (3.0)

$$(3.0) \quad h_2 = 450 - 10x_1 - 15(x_2 = 20 - ({}^1/_4)x_1 - ({}^1/_{20})h_1)$$

$$= 150 - ({}^{25}/_4)x_1 + ({}^3/_4)h_1$$

Substitute the value of x2 in (1.0), the objective function

$$(1.0) \quad z = 45x_1 + 80(20 - ({}^1/_4)x_1 - ({}^1/_{20})h_1) + 0h_1 + 0h_2$$

$$= 1600 + 25x_1 + 0x_2 - 4h_1 + 0h_2$$

$$(1.0) \quad Max\ z = 45x_1 + 80x_2 + 0h_1 + 0h_2$$

$$(2.0) \quad h_1 = 400 - 5x_1 - 20x_2$$

$$(3.0) \quad h_2 = 450 - 10x_1 - 15x_2$$

$$x_1, x_2, h_1, h_2 \geq 0$$

Linear Programming/Simplex Method .. 7

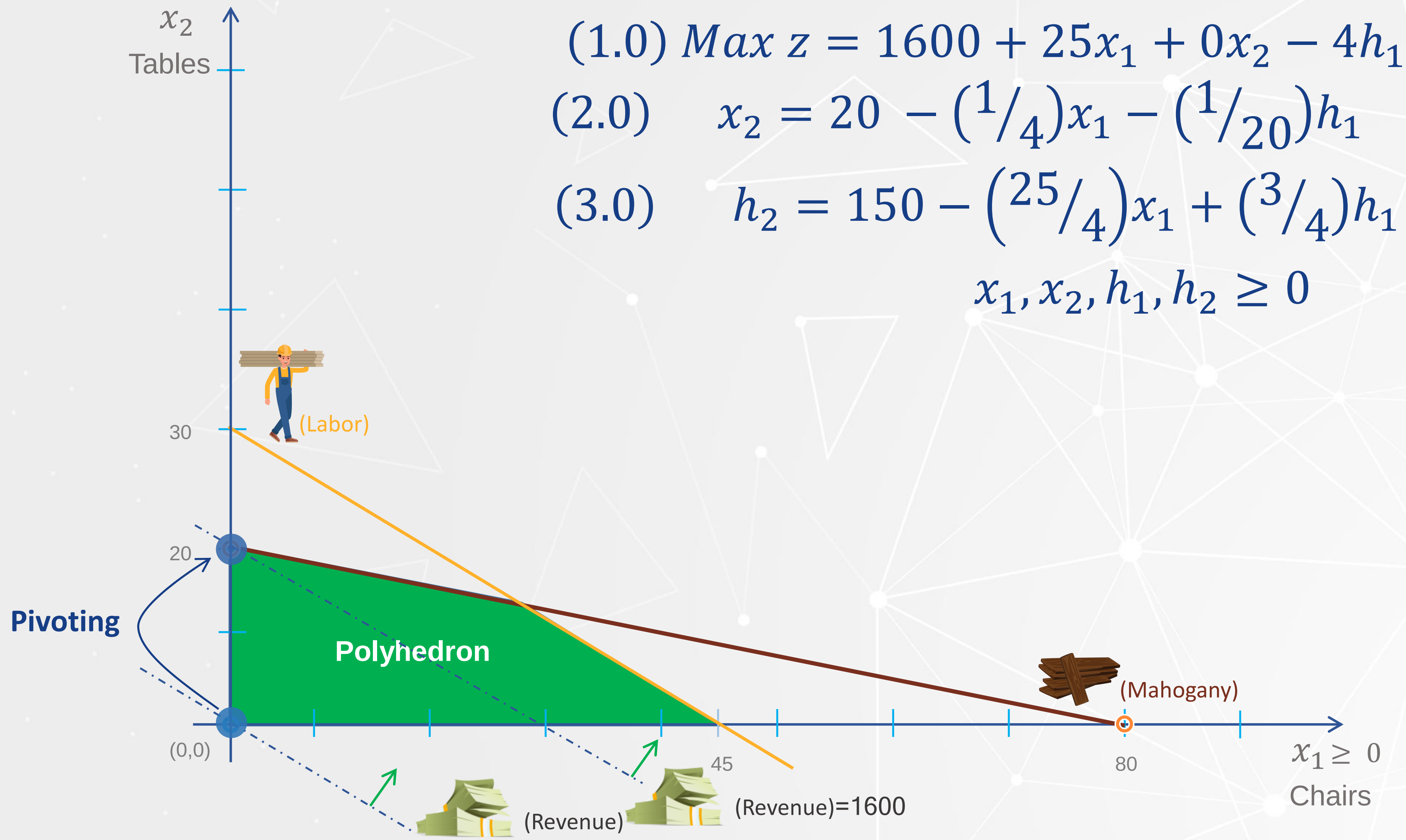
Furniture LP problem in a canonical form with respect to the basic variables (x2, h2).

$$(1.0)\quad Max\ z = 1600 + 25x_1 + 0x_2 - 4h_1 + 0h_2$$

$$(2.0)\quad x_2 = 20 - (1/4)x_1 - (1/20)h_1$$ Production of tables

$$(3.0)\quad h_2 = 150 - (25/4)x_1 + (3/4)h_1$$ Unused labor capacity

$$x_1, x_2, h_1, h_2 \geq 0$$ Non-negativity



Linear Programming/Simplex Method (Pivoting) .. 8

$$(1.0)\ Max\ z = 1600 + 25x_1 + 0x_2 - 4h_1 + 0h_2$$

$$(2.0)\quad x_2 = 20 - (1/4)x_1 - (1/20)h_1$$ Production of tables

$$(3.0)\quad h_2 = 150 - \boxed{(25/4)}x_1 + (3/4)h_1$$ Unused labor capacity

$$x_1, x_2, h_1, h_2 \geq 0$$ Non-negativity

Simplex method: **iteration 2**

Step1: x1 enters the basis.

Step2: minimum ratio test, min{20/(1/4)=80 ,150/(25/4)=24 }=24. h2 leaves the basis

Step3: Pivoting express problem in canonical form with respect to (x2, x1)

Linear Programming/Simplex Method (Pivoting) .. 9

In equation 3.0, express x1 in terms of h1 and h2

$$(3.0) \quad x_1 = 24 + (3/25)h_1 - (4/25)h_2$$

We substitute the value of x1 in equation (2.0)

$$(2.0) \quad x_2 = 14 - \left({}^{2}/_{25}\right)h_1 + \left({}^{1}/_{25}\right)h_2$$

Substitute the value of x1 in (1.0), the objective function

$$z = 2200 + 0x_1 + 0x_2 - 1h_1 - 4h_2$$

$$(1.0) \; Max\ z = 1600 + 25x_1 + 0x_2 - 4h_1 + 0h_2$$

$$(2.0) \quad x_2 = 20 - \left({}^{1}/_{4}\right)x_1 - \left({}^{1}/_{20}\right)h_1$$

$$(3.0) \quad h_2 = 150 - \left({}^{25}/_{4}\right)x_1 + \left({}^{3}/_{4}\right)h_1$$

$$x_1, x_2, h_1, h_2 \geq 0$$

Linear Programming/Simplex Method (Pivoting) .. 10

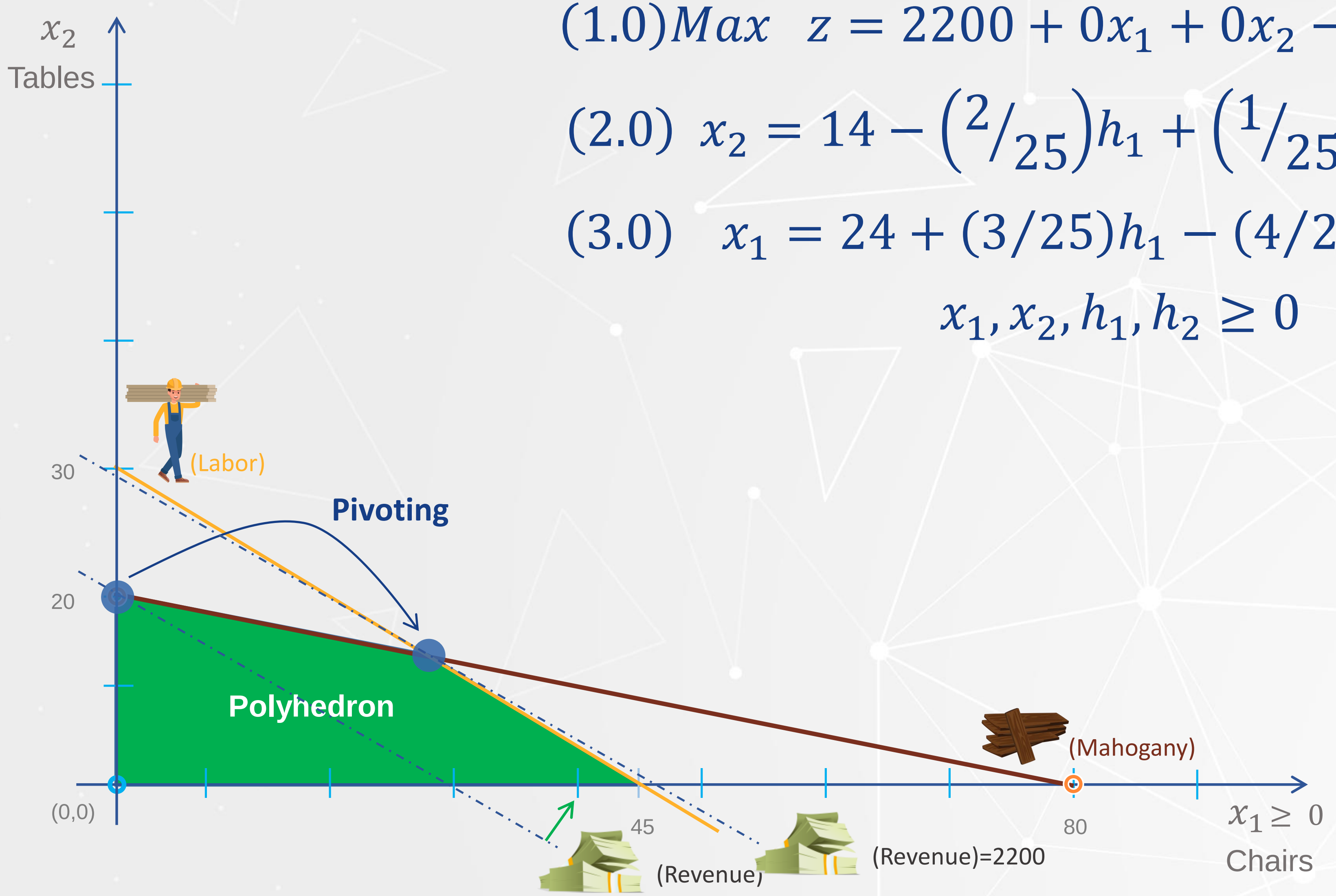
Furniture LP problem in a canonical form with respect to the basic variables (x2, x1).

$$(1.0)\, Max \quad z = 2200 + 0x_1 + 0x_2 - 1h_1 - 4h_2$$

$$(2.0) \quad x_2 = 14 - \left({}^{2}/_{25}\right)h_1 + \left({}^{1}/_{25}\right)h_2$$ Production of tables

$$(3.0) \quad x_1 = 24 + (3/25)h_1 - (4/25)h_2$$ Production of chairs

$$x_1, x_2, h_1, h_2 \geq 0$$ Non-negativity

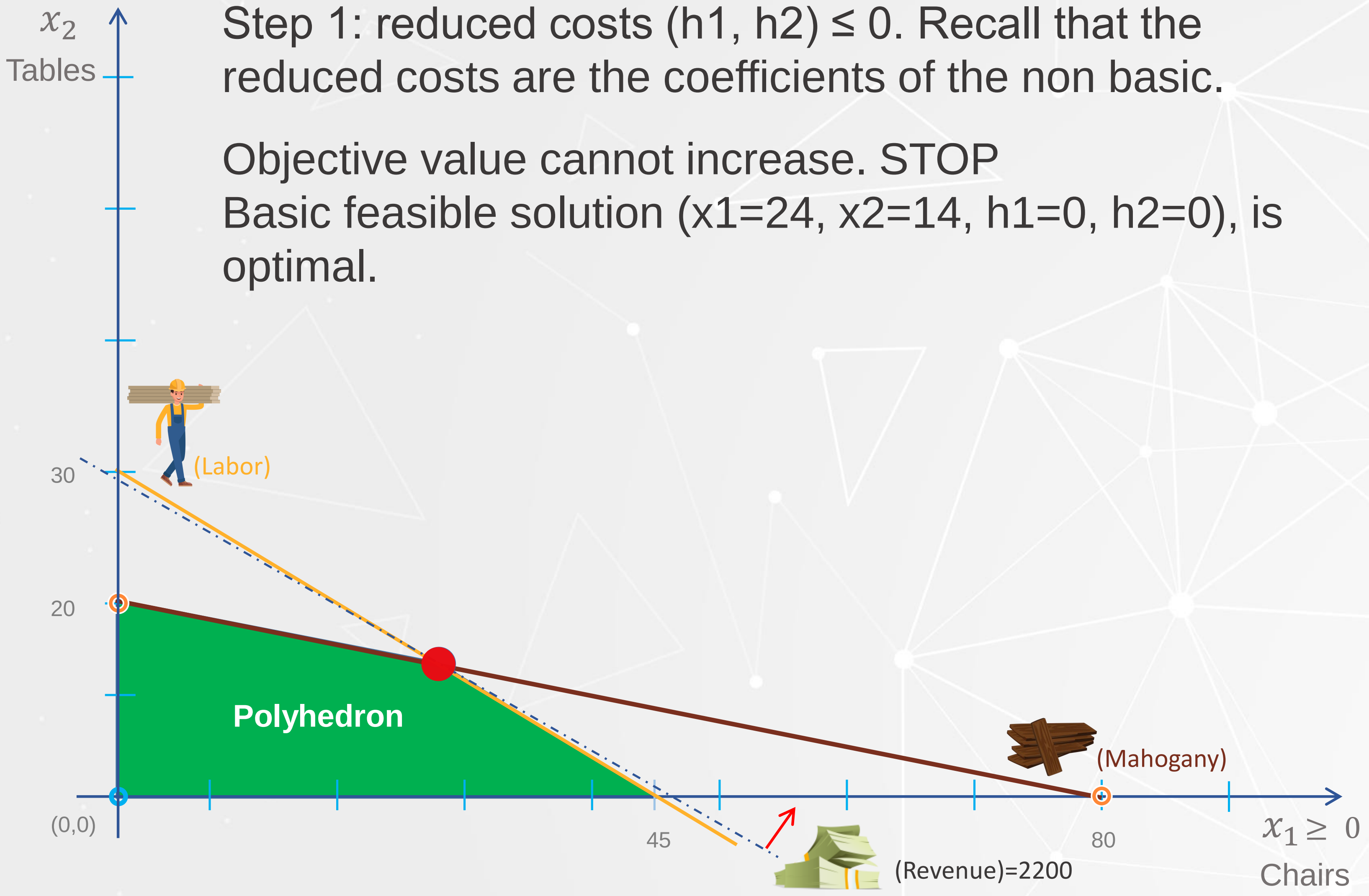


Linear Programming/Simplex Method.. 11

Simplex method: iteration 3.

Step 1: reduced costs (h1, h2) ≤ 0. Recall that the reduced costs are the coefficients of the non basic.

Objective value cannot increase. STOP
Basic feasible solution (x1=24, x2=14, h1=0, h2=0), is optimal.



$$(1.0)\, Max \quad z = 2200 + 0x_1 + 0x_2 - 1h_1 - 4h_2$$

$$(2.0) \quad x_2 = 14 - \left({}^{2}/_{25}\right)h_1 + \left({}^{1}/_{25}\right)h_2$$

$$(3.0) \quad x_1 = 24 + (3/25)h_1 - (4/25)h_2$$

$$x_1, x_2, h_1, h_2 \geq 0$$

Summary of simplex method for the maximization case

1. Transform the original LP problem into the standard form. Consider an initial basic feasible solution.

2. Express the LP problem in a canonical form with respect to the current basic feasible solution.

3. If the reduced costs of all the non basic variables are ≤ 0, **STOP** –the current basic feasible solution is optimal. Else, choose a non basic variable with the largest positive reduced cost to enter the basis.

4. Consider the column vector of the non basic variable entering the basis. If all the coefficients of this column vector are positive, the entering non basic variable can be arbitrarily large, hence the LP problem is unbounded.
   i. Assume that the column vector has at least one negative component.
   ii. Apply the minimum ratio test over the equations where the entering non basic variable has negative coefficients to determine the basic variable that will leave the basis.
   iii. (Pivoting) Go to 2.) to determine the new basic solution.

GUROBI OPTIMIZATION

Modeling and solving LP problems

Gurobi Python API

Furniture Problem: solved with Gurobi … 1

Manufacturing Problem: Furniture factory

$$
\begin{aligned}
&\textit{Max Revenue} = 45x_1 + 80x_2 \\
&\textit{Subject to}: \\
&\textit{Mahogany}: 5x_1 + 20x_2 \le 400 \\
&\textit{Labor}: 10x_1 + 15x_2 \le 450 \\
&\textit{Non-negativity}: x_1, x_2 \ge 0
\end{aligned}
$$

```python
# import gurobi library
from gurobipy import *
```

This command imports the Gurobi functions and classes.

```python
# Create Furniture new model
f = Model("Furniture")
```

The Model() constructor creates a model object **f** . The name of this new model is ‘Furniture’. This new model **f** initially contains no decision variables, constraints, or objective function.

```python
# Create variables
x1 = f.addVar(name="chairs")
x2 = f.addVar(name="tables")
```

This method adds a decision variable to the model object **f**, one by one; i.e. x1 and then x2. The argument of the method gives the name of added decision variable. The default values are applied here; i.e. the decision variables are of type continuous and non-negative, with no upper bound.

```python
# Define objective function
f.setObjective(45*x1 + 80*x2, GRB.MAXIMIZE)
```

This method adds the objective function to the model object **f**. The first argument is a linear expression (LinExpr) and the second argument defines the sense of the optimization.

A linear expression object (LinExpr) consists of a constant term, plus a sum of coefficient-variables pairs that capture the linear terms.

```python
# Add mahogany constraint
f.addConstr(5*x1 + 20*x2 <= 400, "mahogany")
# add labor constraint
f.addConstr(10*x1 + 15*x2 <= 450, "labor")
```

This method adds a constraint to the model object f and considers a linear expression (LinExpr) as the left-hand-side of the constraints, the sense of the constraint, and its capacity value. The last argument gives the name of the constraint.

Furniture Problem: solved with Gurobi … 2

```
# Run optimization engine
f.optimize()
```

This method runs the optimization engine to solve the LP problem in the model object f

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 2e+01]
  Objective range  [5e+01, 8e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+02]
Presolve time: 0.01s
Presolved: 2 rows, 2 columns, 4 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    6.5000000e+31   2.968750e+30   6.500000e+01      0s
       2    2.2000000e+03   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.02 seconds
Optimal objective  2.200000000e+03
```

Minimum and maximum absolute value of the matrix of technology coefficients.
Minimum and maximum absolute value of the objective function coefficients.
Minimum and maximum absolute value of the upper and lower bound values.
Minimum and maximum absolute value of the RHS values.

Simplex method iteration information.

An optimal solution was found.

```
# Display optimal production plan
for v in f.getVars():
    print(v.varName, v.x)

print('Optimal total revenue:', f.objVal)
```

This method retrieves a list of all variables in the model object **f**

The print function displays the decision variable names and solution value

The print function displays the objective function value of the model object **f**

```
('chairs', 24.0)
('tables', 14.0)
('Optimal total revenue:', 2200.0)
```

Optimal Production Plan

What if our LP problem has hundreds of thousands variables and constraints?

- The Gurobi python code just presented is too manual and would take too long to build a large scale LP problem.

- We should use appropriate data structures and Gurobi python functions and objects to abstract the problem, and have the Gurobi python code build the LP problem of any size.

Furniture Problem (abstracted): solved with Gurobi .. 1

General Furniture model formulation

Let $price_p$ be the price of product $p \in products = \{chairs,\ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resource = \{mahogany,\ labor\}$.

Let $bom_{r,j}$ be the amount of resource r required by product p. Then the general formulation of the Furniture problem is:

Parametrized furniture LP problem formulation

$$Max \sum_{p \in products} price_p make_p$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p \leq capacity_r \quad \forall\, r \in resources$$

$$make_p \geq 0 \quad \forall\, p \in products$$

Furniture Problem: Parametrized solved with Gurobi .. 2

```
# import gurobi library
from gurobipy import *
```

```
# resources data
resources, capacity = multidict({
  'mahogany': 400,
  'labor':  450 })
```

The multidict function returns a list which maps each resource (key) to its capacity value.

```
# products data,
products,  price  = multidict({
  'chair': 45,
  'table':  80 })
```

This multidict function returns a list which maps each product (key) to its price value.

```
# Bill of materials: resources required by each product
bom = {
  ('mahogany', 'chair'):  5,
  ('mahogany', 'table'): 20,
  ('labor',    'chair'): 10,
  ('labor',    'table'): 15 }
```

This dictionary has a 2-tuple as a key, mapping the resource required by a product with its quantity per.

Furniture Problem: Parametrized solved with Gurobi .. 3

```
# Declare and initialize model
f = Model('Furniture')
```

The Model() constructor creates a model object f .

```
# Create decision variables for the products to make
make = f.addVars(products, name="make")
```

This method adds decision variables to the model object f, and returns a Gurobi tupledict object (make) that contains the variables recently created.

The first argument (products) provides the indices that will be used as keys to access the variables in the returned tupledict. The last argument gives the name ‘make’ to the decision variables. The decision variables are of type continuous and non-negative, with no upper bound.

Furniture Problem: Parametrized solved with Gurobi .. 4

This method adds constraints to the model object f.

$$\sum_{p \in products} bom_{r,p} make_p \leq capacity_r \quad \forall\, r \in resources$$

```
res = f.addConstrs(((sum(bom[r,p]*make[p] for p in products) <= capacity[r]) for r in resources), name='R')
```

Furniture Problem: Parametrized solved with Gurobi .. 5

```
# The objective is to maximize total profit
f.setObjective(make.prod(revenue), GRB.MAXIMIZE)
```

$$Max \sum_{p \in products} price_p make_p$$

This method adds the objective function to the model object f. The first argument is a linear expression which is generated by the (prod) method. The (prod) method is the product of the object (revenue) with the object (make) for each product p in the set (products). The second argument defines the sense of the optimization.

Furniture Problem: Parametrized solved with Gurobi .. 6

```python
# save model for inspection
f.write('furniture.lp')
```

```
\ Model Furniture
\ LP format - for model browsing. Use MPS format to capture full model detail.
Maximize
  80 make[table] + 45 make[chair]
Subject To
 R[mahogany]: 20 make[table] + 5 make[chair] <= 400
 R[labor]: 15 make[table] + 10 make[chair] <= 450
Bounds
End
```

Furniture Problem: Parametrized solved with Gurobi .. 7

```
# run optimization engine
f.optimize()
```

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 2e+01]
  Objective range  [5e+01, 8e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+02]
Presolve time: 0.02s
Presolved: 2 rows, 2 columns, 4 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    6.5000000e+31   2.968750e+30   6.500000e+01      0s
       2    2.2000000e+03   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.02 seconds
Optimal objective  2.200000000e+03
```

```
# display optimal values of decision variables
for v in f.getVars():
    if (abs(v.x) > 1e-6):
        print(v.varName, v.x)

# display optimal total profit value
print('total revenue', f.objVal)
```

```
('make[table]', 14.0)
('make[chair]', 24.0)
('total revenue', 2200.0)
```

This method runs the optimization engine to solve the LP problem in the model object f

Optimal Production Plan

Final Remarks

- The simplex method iterates from a basic feasible solution to another, always trying to improve the value of the solution, until further improvement cannot be done.
- The geometric interpretation of a basic feasible solution is a corner point (vertex) of the polyhedron that determines the feasible region of an LP problem.

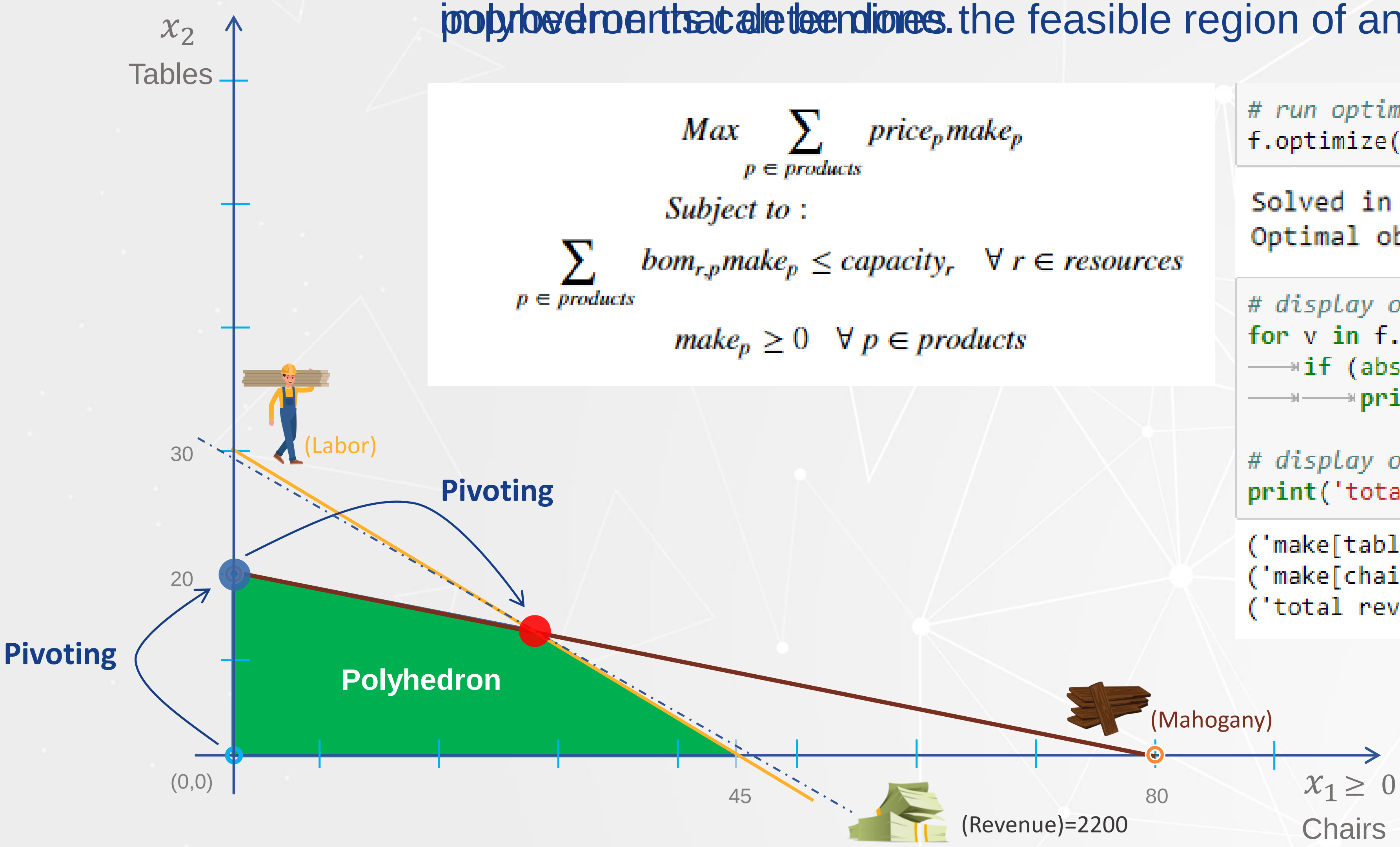
$$\begin{aligned} Max \sum_{p \in products} price_p make_p \\ Subject\ to: \\ \sum_{p \in products} bom_{r,p} make_p \leq capacity_r \quad \forall\, r \in resources \\ make_p \geq 0 \quad \forall\, p \in products \end{aligned}$$

```python
# run optimization engine
f.optimize()
```

```
Solved in 2 iterations and 0.02 seconds
Optimal objective  2.200000000e+03
```

```python
# display optimal values of decision variables
for v in f.getVars():
    if (abs(v.x) > 1e-6):
        print(v.varName, v.x)

# display optimal total profit value
print('total revenue', f.objVal)
```

```
('make[table]', 14.0)
('make[chair]', 24.0)
('total revenue', 2200.0)
```



Sensitivity analysis of LP problems

Gurobi Python API

Furniture Problem: economic interpretation

Economic interpretation in Linear Programming models

- Solving LP problems provides more information than only the values of the decision variables and the value of the objective function.

- Associated with an LP optimal solution there are ***shadow prices*** (a.k.a. ***dual variables***, or ***marginal values***) for the constraints.

- The shadow price of a constraint associated with the optimal solution, represents the change in the value of the objective function per unit of increase in the right-hand side value of that constraint.

- There are shadow prices associated with the non-negativity constraints. These shadow prices are called the ***reduced costs***.

Furniture Problem: economic interpretation .. 2

- For example, suppose the labor capacity is increased from 450 hours to 451 hours. What is the increase in the objective function value from such increase?
- Since the constraints on mahogany capacity (2.0) and labor capacity (3.0) define the optimal solution, we can solve the following system of equations

$$5x_1+20x_2 = 400 \quad \text{Mahogany capacity}$$
$$10x_1 + 15x_2 = 451 \quad \text{Labor capacity}$$

- The new values of the decision variables are: chairs (x1) = 24.16, tables (x2) = 13.96
- The new value of the objective function (revenue) is = $2,204
- The shadow price associated with the labor capacity is $2,204 - $2,200 = $4. That is, we can get $4 of increased revenue per hour of increase in labor capacity.
- **Remark:** The shadow price value of $4 remains constant over a range of value changes of the mahogany capacity. The calculation of this range is beyond the scope of this course.

Furniture Problem: economic interpretation .. 3

- Similarly, we can compute the shadow price of the mahogany constraint by solving the following system of equations

$$5x_1 + 20x_2 = 401 \quad \text{Mahogany capacity}$$
$$10x_1 + 15x_2 = 450 \quad \text{Labor capacity}$$

- The new values of the decision variables are: chairs (x1) = 14.08, tables (x2) = 23.88
- The new value of the objective function (revenue) is = $2,201
- The shadow price associated with the mahogany capacity is $2,201 - $2,200 = $1. That is, we can get $1 of increased revenue per unit of increase in mahogany capacity.
- **Remark:** The shadow price value of $1 remains constant over a range of value changes of the labor capacity.

GUROBI OPTIMIZATION

Furniture Problem: simplex method revisited

The LP problem in a canonical form with respect to the optimal basic variables (x1, x2) is

$$Max \quad z = 2200 + 0x_1 + 0x_2 - 1h_1 - 4h_2$$

$$(2.0) \quad x_2 = 14 - \left({}^{2}/_{25}\right)h_1 + \left({}^{4}/_{25}\right)h_2$$

$$(3.0) \quad x_1 = 24 + (3/25)h_1 - (4/25)h_2$$

$$x_1, x_2, h_1, h_2 \geq 0$$

- **Remarks**: Recall that h1 and h2 are the slack variables associated with the mahogany capacity constraint and labor capacity constraint, respectively.
- The interpretation of the slack variables is the amount of resource capacity not consumed by the production of chairs and tables.
- The optimal solution is
  - Revenue = $2,200, chairs (x1) = 24, tables (x2)= 14
  - Slack variable (h1) of mahogany constraint = 0, and
  - Slack variable (h2) of labor constraint = 0
- **Remarks**: If we increase the value of h1 (unused capacity of mahogany) by one the total revenue will be reduced by $1. If we increase the value of h2 (unused capacity of labor) by one the total revenue will be reduced by $4.
- Our previous analysis show us that the shadow price of mahogany is $1 and the one of labor is $4 !!!!
- **Conclusion**: The simplex method automatically give us the shadow prices of the resources.

Furniture Problem: solved with Gurobi

This method adds constraints to the model object f. We store the constraints generated in an object called (res).

```
res = f.addConstrs(((sum(bom[r,p]*make[p] for p in products) <= capacity[r]) for r in resources), name='R')
```

For each resource constraint in the dictionary (res), check if its associated shadow price is greater than zero. Then print the resource constraint name and the resource constraint shadow price.

Recall that the object (res) stores all the information related to the constraints of the model f.

```
# display shadow prices of resources constraints
for r in res:
    if (abs(res[r].Pi) > 1e-6):
        print(res[r].ConstrName, res[r].Pi)
```

```
('R[mahogany]', 1.0)
('R[labor]', 4.0)
```

Furniture Problem: economic interpretation .. 4

- **Is it profitable to make a third product, like desks?**
  - **Assume that the price of the desk is $110,**
  - **and the desk consumes 15 units of mahogany and 25 units of labor**

    - The previous python code has parametrized the Furniture LP model, i.e. the model formulation does not depend on the data of the problem. Therefore, we just generate a new set of data that includes the new product information

```
# products data,
products, price  = multidict({
    'chair': 45,
    'table':  80,
    'desk': 110 })
```

```
# Bill of materials: resources required by each product
bom = {
    ('mahogany', 'chair'): 5,
    ('mahogany', 'table'):  20,
    ('mahogany', 'desk'): 15,
    ('labor', 'chair'):     10,
    ('labor', 'table'):   15,
    ('labor', 'desk'):   25 }
```

Furniture Problem: economic interpretation .. 5

- The new LP model is

```python
# save model for inspection
f.write('furniture.lp')
```

```
\ Model Furniture
\ LP format - for model browsing. Use MPS format to capture full model detail.
Maximize
  80 make[table] + 45 make[chair] + 110 make[desk]
Subject To
 mahogany: 20 make[table] + 5 make[chair] + 15 make[desk] <= 400
 labor: 15 make[table] + 10 make[chair] + 25 make[desk] <= 450
Bounds
End
```

Furniture Problem: economic interpretation .. 6

```
# run optimization engine
f.optimize()
```

```
# display optimal values of decision variables
for v in f.getVars():
    if (abs(v.x) > 1e-6):
        print(v.varName, v.x)

# display optimal total profit value
print('total profits', f.objVal)
```

```
('make[table]', 14.0)
('make[chair]', 24.0)
('total profits', 2200.0)
```

It is not profitable to produce desks. The optimal solution remains the same.

```
# display shadow prices of resources constraints
for r in res:
    if (abs(res[r].Pi) > 1e-6):
        print(res[r].ConstrName, res[r].Pi)
```

```
('R[mahogany]', 1.0)
('R[labor]', 4.0)
```

The shadow prices of the resources remain the same.

Furniture Problem: economic interpretation .. 7

- Notice that we can use the shadow price information of the resources to check if it is worth it to make desks.
  - The shadow price of the mahogany capacity constraint is $1
  - The shadow price of the labor capacity constraint is $4
  - Let's compute the opportunity cost of making one desk and compare it with the price of a desk. If this opportunity cost is greater than the price, then it is not worth it to make desks.
  - The opportunity cost can be computed by multiplying the units of mahogany capacity that one desk built consumes by the shadow price of mahogany capacity, and multiplying the hours of labor capacity that one desk built consumes by the shadow price of labor capacity:
    - That is, ($1)*15 (units of mahogany) + ($4)*25 (hours of labor) = $115 > $110
- Therefore, investing resources to produce desks, otherwise used to produce chairs and tables, is not profitable.

Multiple optimal solutions

Gurobi Python API

GUROBI OPTIMIZATION

Furniture Problem

New business conditions

- The data scientist in charge of production planning for the furniture factory uses machine learning to predict that the market price of a chair is now \$50 and the market price of a table is \$75.
- The equation of the objective function under these new business conditions is: revenue = 50x1 + 75x2.

GUROBI OPTIMIZATION

Furniture Problem

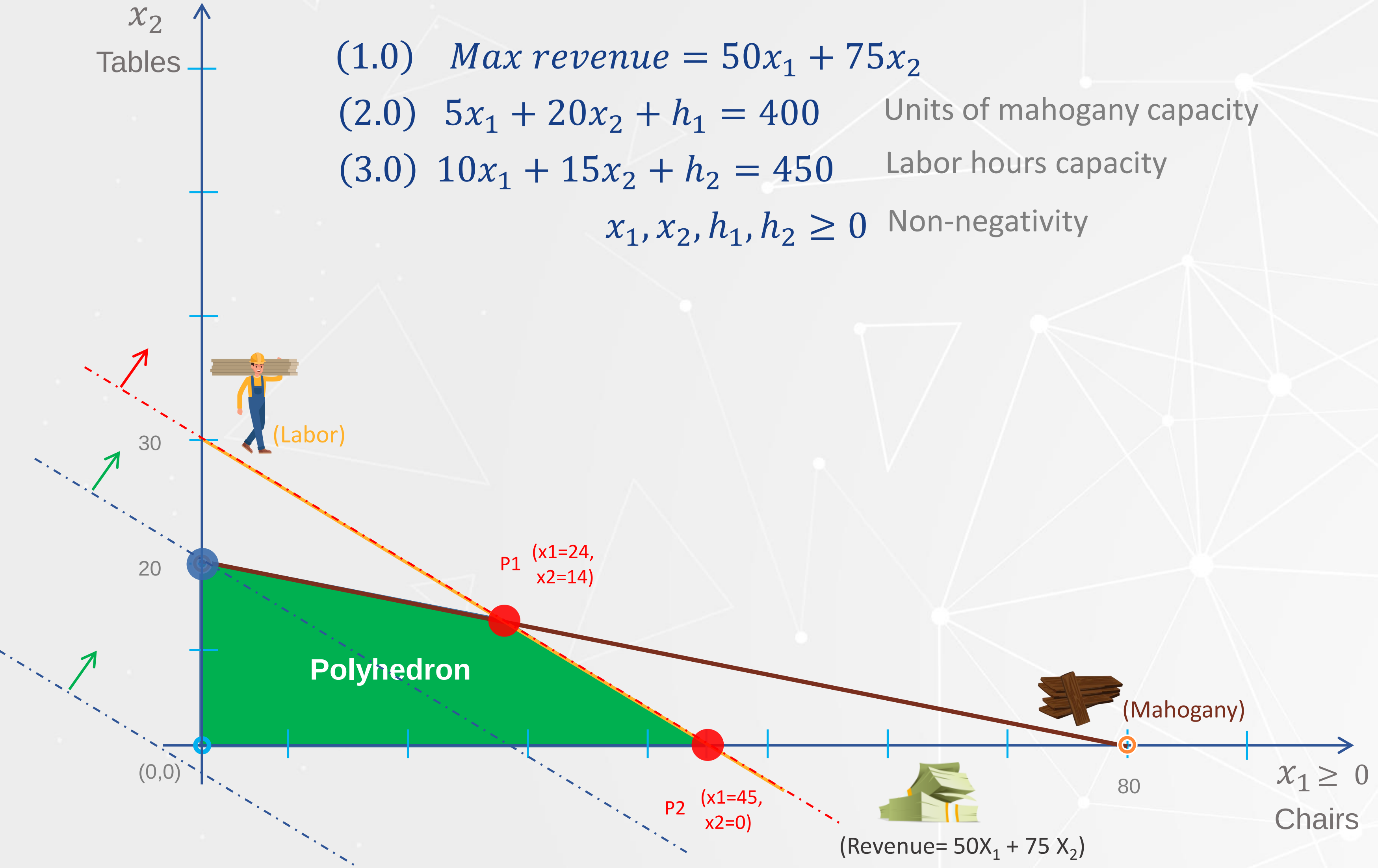
Furniture problem standard form

$$(1.0)\quad Max\ revenue = 50x_1 + 75x_2$$

$$(2.0)\quad 5x_1 + 20x_2 + h_1 = 400$$ Units of mahogany capacity

$$(3.0)\quad 10x_1 + 15x_2 + h_2 = 450$$ Labor hours capacity

$$x_1, x_2, h_1, h_2 \geq 0$$ Non-negativity



New business conditions

- In this case, the production plan P1 of building 24 chairs and 14 tables and the plan P2 of building 45 chairs and 0 tables are both optimal.
- The production plan P1 is defined by the mahogany and labor constraints. The associated optimal basic feasible solution is x1=24, x2=14, h1=0, h2=0.
- The production plan P2 is defined by the labor and a non-negativity constraint (x2 = 0), i.e. no tables are built. The associated optimal basic feasible solution is x1=45, x2=0, h1=175, h2=0.

Modified Furniture Problem: solved with Gurobi … 1

General Furniture model formulation -alternative optimal solutions

Let $price_p$ be the price of product $p \in products = \{chairs, \ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resources = \{mahogany, \ labor\}$.

Let $bom_{r,j}$ be the amount of resource r required by product p. Then the general formulation of the Furniture problem is:

Parametrized furniture LP problem formulation

$$Max \sum_{p \in products} price_p make_p$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p \le capacity_r \quad \forall\ r \in resources$$

$$make_p \ge 0 \quad \forall\ p \in products$$

We modify the objective function as follows:

New	Price
Chairs	$50
Tables	$75

```python
# products data.
products, price = multidict({
  'chair': 50,
  'table':  75 })
```

New data, same model formulation

Modified Furniture Problem: solved with Gurobi … 2

```
# run optimization engine
f.optimize()

# display optimal values of decision variables
for v in f.getVars():
    print(v.varName, v.x)

# display optimal total profit value
print('total revenue', f.objVal)
```

```
('make[table]', 0.0)
('make[chair]', 45.0)
('total revenue', 2250.0)
```

```
# display shadow prices of resources constraints
for r in res:
    print(res[r].ConstrName, res[r].Pi)
```

```
('R[mahogany]', 0.0)
('R[labor]', 5.0)
```

```
# display reduced costs of decision variables
for v in f.getVars():
    print(v.varName, abs(v.rc))
```

```
('make[table]', 0.0)
('make[chair]', 0.0)
```

Gurobi found one of the optimal solutions ... P2

Notice that the shadow price of the resource mahogany for this alternative optimal solution is zero,
… the marginal value of mahogany for this optimal solution is zero.

Reduced cost of the non basic variable (tables) is zero. This means that if we produce more tables (and produce less chairs) the revenue generated remains the same.

Furniture Problem: simplex method revisited .. 1

New (original) LP problem

$$Max\ z = 50x1 + 75x2$$
$$s.t.\ 5x1 + 20x2 \leq 400$$
$$10x1 + 15x2 \leq 450$$
$$x1, x2 \geq 0$$

LP problem in canonical form with respect to the optimal basic variables (x1, h1) found by Gurobi:

$$Max\ z = 2{,}250 + (0x1 + 0h1) + (0x2 - 5h2)$$

Non basic variables

$$h1 = 175 - \left({}^{25}/_{2}\right)x2 + \left({}^{1}/_{2}\right)h2 \quad (2.0)$$
$$x1 = 45 - \left({}^{3}/_{2}\right)x2 - \left({}^{1}/_{10}\right)h2 \quad (3.0)$$
$$x1, x2, h1, h2 \geq 0$$

The LP problem in a standard form is

$$Max\ z = 50x1 + 75x2 + 0h1 + 0h2$$

$$s.t.\ 5x1 + 20x2 + h1 = 400$$
$$10x1 + 15x2 + h2 = 450$$
$$x1, x2, h1, h2 \geq 0$$

The reduced cost of non basic variable x2 is zero, hence if we increase its value, the optimal objective function value does not change. Let’s decide which basic variable should become non basic (value of zero) by computing the minimum ratio test: $min\{175/(25/2) = 14\,, 45/(3/2) = 30\} = 14;$ therefore h1 will become non basic variable. Hence, we pivot on constraint (2.0) and the column of variable x2.

Furniture Problem: simplex method revisited .. 2

LP problem in canonical form with respect to the basic variables (x1, x2):

$$Max\ z = 2{,}250 + 0h1 - 5h2$$

$$x2 = 14 - \left({}^{2}/_{25}\right)h1 + \left({}^{1}/_{25}\right)h2 \quad (2.0)$$
$$x1 = 24 - \left({}^{3}/_{25}\right)h1 - \left({}^{4}/_{25}\right)h2 \quad (3.0)$$
$$x1, x2, h1, h2 \geq 0$$

Modeling Opportunity

From the mathematical model point of view, we have two alternative optimal solutions S1= (x1 =24, x2 =14, h1=0, h2= 0) and S2 = (x1 =45, x2 =0, h1= 175, h2 = 0), both with a maximum total revenue of $2,250.

The data scientist points out that from the business perspective, the “optimal” solution S1 is preferred to the “optimal” solution S2, because the latter solution wastes 175 units of mahogany.

The data scientist decides to modify the LP model by now interpreting the slack variables as the amount of wasted resources, and defines the new decision variables:
x3 is the amount of unused mahogany and x4 is the amount of unused labor.

The data scientists uses machine learning to predict that the per unit inventory carrying cost of mahogany is $1 , and the unused per hour labor cost is $2. Then, the new LP model formulation is:

Furniture Problem: simplex method revisited .. 2

Modeling Opportunity
From the mathematical model point of view, we have two alternative optimal solutions S1= (x1 =24, x2 =14, x3 = x4 = 0) and S2 = (x1 =45, x2 =0, x3 = 175, x4 = 0), both with a maximum total revenue of $2,250.

The data scientist notice that from the business perspective, the "optimal" solution S1 is preferred to "optimal" solution S2, because with the latter solution is wasting 175 units of mahogany.

The data scientist decides to modify the LP model by now interpreting the slack variables as the amount of wasted resources, and defines the new decision variables:
x3 is the amount of unused mahogany and x4 is the amount of unused labor.

The data scientists using machine learning estimates that the per unit inventory carrying cost of mahogany is $1 , and the unused per hour labor cost is $2. Then, the new LP model formulation is:

$$Max\ z = 50x1 + 75x2 - 1x3 - 2x4$$

$$s.t.\ 5x1 + 20x2 + x3 = 400$$
$$10x1 + 15x2 + x4 = 450$$
$$x1, x2, x3, x4 \geq 0$$

New Furniture Problem: solved with Gurobi … 1

General Furniture model formulation -Penalize wasting resources

Let $price_p$ be the price of product $p \in products = \{chairs, \ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resource = \{mahogany, \ labor\}$.

Let $waste_r$ be a new decision variable that measures the amount of unused resource $r \in resoruces = \{mahogany, \ labor\}$. Let $cost_r$ be the per unit cost of unused resource capacity.

Let $bom_{r,j}$ be the amount of resource r required by product p. Then the new formulation of the Furniture problem is:

$$Max \sum_{p \in products} price_p make_p - \sum_{r \in resources} cost_r waste_r$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p + waste_r = capacity_r \quad \forall\, r \in resources$$

$$make_p \geq 0 \quad \forall\, p \in products$$

$$waste_r \geq 0 \quad \forall\, r \in resources$$

Price

Per unit	Price
Chairs	\$50
Tables	\$75

Resources cost table:

Per unit	Cost
Mahogany	\$1
Labor	\$2

New Furniture Problem: solved with Gurobi … 2

```
# import gurobi library
from gurobipy import *
```

```
# resources data
resources, capacity, cost = multidict({
  'mahogany': [400, 1],
  'labor':  [450, 2] })
```

We add a cost parameter to the resources multidict to penalize the waste of resources.

```
# products data,
products,  price = multidict({
  'chair': 50,
  'table':  75 })
```

```
# Bill of materials: resources required by each product
bom = {
  ('mahogany', 'chair'): 5,
  ('mahogany', 'table'):  20,
  ('labor', 'chair'):      10,
  ('labor', 'table'):   15 }
```

New Furniture Problem: solved with Gurobi … 3

```python
# Declare and initialize model
f = Model('Furniture')
```

```python
# Create decision variables for the products to make
make = f.addVars(products, name="make")
```

```python
# Create decision variables for the wasted resources
waste = f.addVars(resources, name="waste")
```

We add a new type of decision variable to measure the unused resource capacity.

```python
# Capacity resource constraints:
# The amount consumed by the products made + the waste = capacity

res = f.addConstrs(((sum(bom[r,p]*make[p] for p in products) + waste[r] == capacity[r]) for r in resources), name='R')
```

$$\sum_{p \in products} bom_{r,p} make_p + waste_r = capacity_r \quad \forall\, r \in resources$$

```python
# The objective is to maximize total profit
f.setObjective(make.prod( price ) - waste.prod(cost) , GRB.MAXIMIZE)
```

$$Max \sum_{p \in products} price_p make_p - \sum_{r \in resources} cost_r waste_r$$

New Furniture Problem: solved with Gurobi … 4

```python
# run optimization engine
f.optimize()
```

```python
# display optimal values of decision variables
for v in f.getVars():
    print(v.varName, v.x)

# display optimal total profit value
print('total profits', f.objVal)
```

```
('make[table]', 14.0)
('make[chair]', 24.0)
('waste[mahogany]', 0.0)
('waste[labor]', 0.0)
('total profits', 2250.0)
```

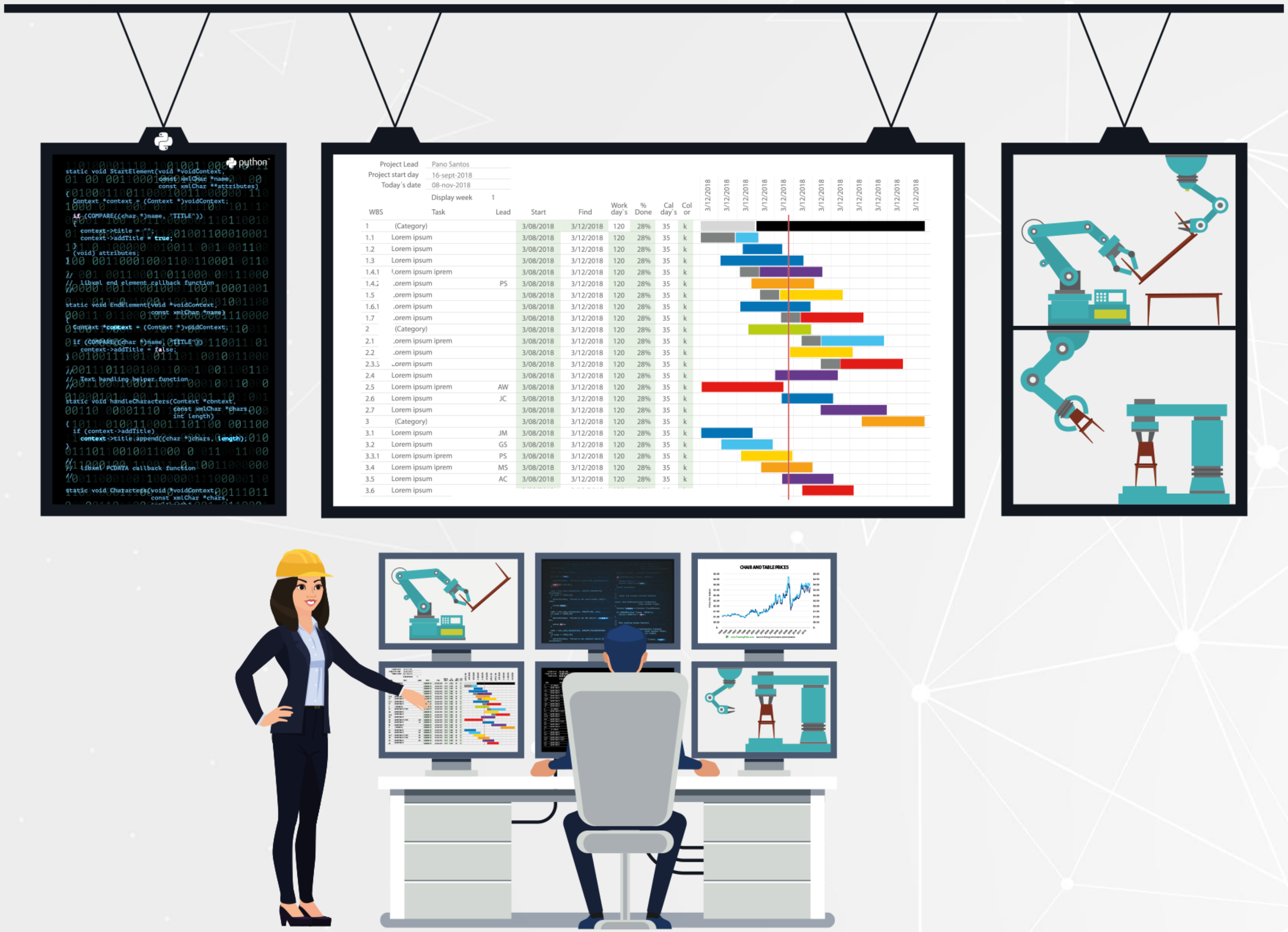
The optimal solution is now making 14 tables and 24 chairs with a total objective function value of $2,250.

Notice that the solution of making 0 tables and 45 chairs is no longer optimal since the objective function value is
($50*45 = $2,250) – ($1*175 = $175) = $2,075.
That is, the alternative solution (chairs = 45, tables =0) is no longer optimal.

Unbounded LP problem

Gurobi Python API

Furniture Problem



New business conditions

- The operations manager of the furniture factory tells the data scientist that she is negotiating with the supplier of mahogany to obtain an unlimited amount of mahogany as long as the factory procures at least 400 units of mahogany per week.
- In addition, the operations manager is negotiating with the union to obtain extra workforce as long as the factory hires at least 450 hours of labor per week.

Furniture Problem

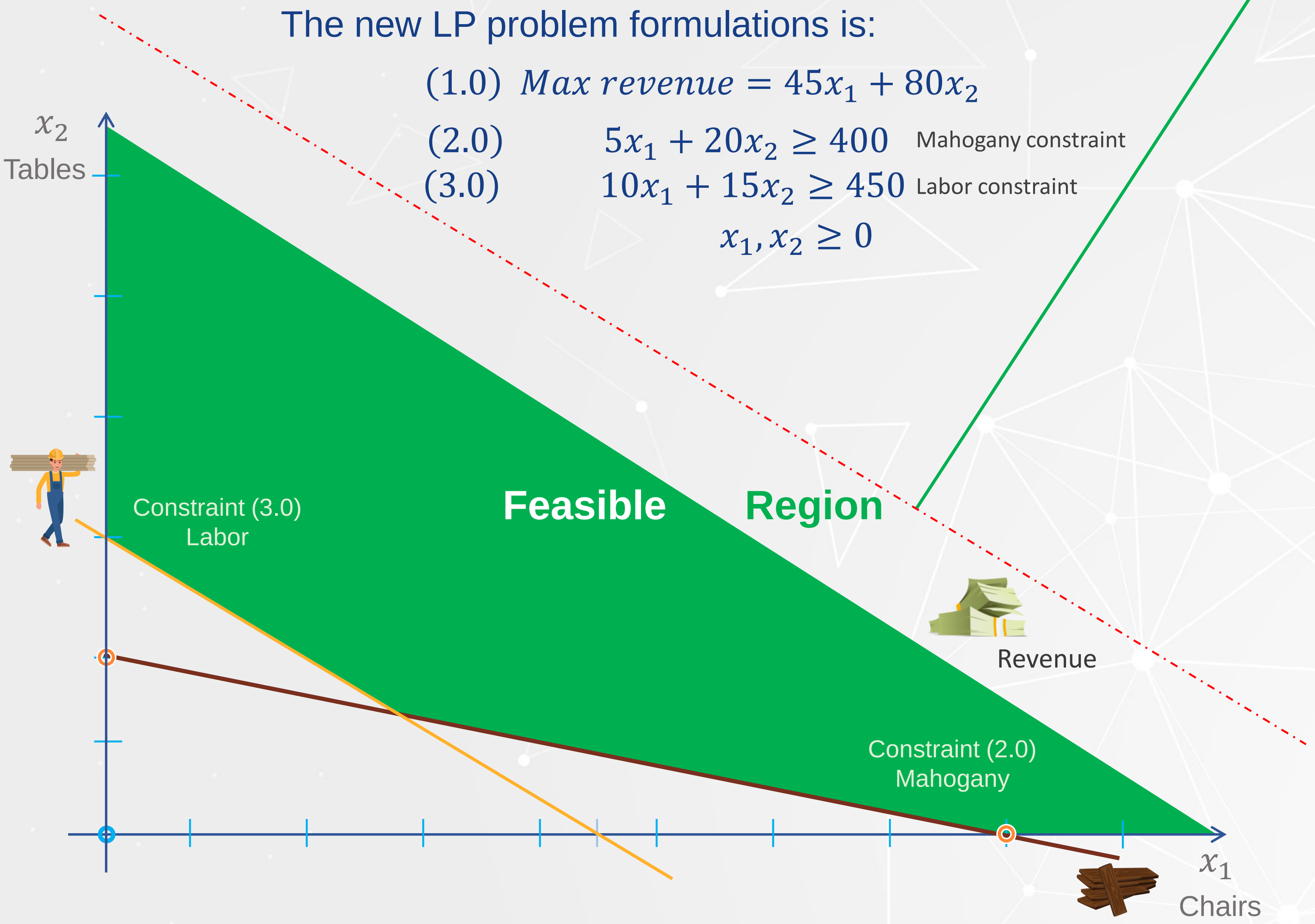
The new LP problem formulations is:

$$(1.0)\quad Max\ revenue = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \geq 400 \quad \text{Mahogany constraint}$$

$$(3.0)\quad 10x_1 + 15x_2 \geq 450 \quad \text{Labor constraint}$$

$$x_1, x_2 \geq 0$$



New business conditions

- Notice that the polyhedron defined by the feasible region is unbounded.
- Clearly, the maximum of the objective function is unbounded, since the revenue that we can make can be arbitrarily large by increasing the number of chairs (x1) and/or the number of tables (x2) as much as we want. Recall, that we had assumed that we sell everything that we produce.
- Conclusion: With unlimited mahogany and labor, revenue is unlimited.

Unbounded Furniture Problem: solved with Gurobi .. 1

```
# Create an object of type list to store
#the constraints for each resource

res = f.addConstrs(((sum(bom[r,p]*make[p] for p in products) >= capacity[r]) for r in resources), name='R')
```

$$\sum_{p \in products} bom_{r,p} make_p \geq capacity_r \quad \forall\, r \in resources$$

```
# run optimization engine
f.optimize()
```

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 2e+01]
  Objective range  [5e+01, 8e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+02]
Presolve time: 0.03s

Solved in 0 iterations and 0.03 seconds
Infeasible or unbounded model
```

The Gurobi solver could not find an optimal solution and declares the problem either infeasible or unbounded.

Unbounded Furniture Problem: solved with Gurobi .. 2

```
print('total profits', f.objVal)
for v in f.getVars():
    print(v.varName, v.x)
```

Since an optimal solution does not exist, the objective function value and the value of the decision variables is void. In this case, we get an error when trying to print these values.

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-11-a1b5e7b87473> in <module>()
----> 1 print('total profits', f.objVal)
      2 for v in f.getVars():
      3     print(v.varName, v.x)

model.pxi in gurobipy.Model.__getattr__ (../../src/python/gurobipy.c:50100)()

model.pxi in gurobipy.Model.getAttr (../../src/python/gurobipy.c:65080)()

model.pxi in gurobipy.Model.__gettypedattr (../../src/python/gurobipy.c:99204)()

AttributeError: Unable to retrieve attribute 'objVal'
```

```
# display optimal values of decision variables
if f.status == GRB.Status.OPTIMAL:
    print('Optimal solution found')
    print('total profits', f.objVal)
    for v in f.getVars():
        print(v.varName, v.x)
```

To avoid the error, we need to check the status of the LP model, and only print the solution values if an optimal solution was found.

Unbounded Furniture Problem: solved with Gurobi .. 3

```
f.setObjective(make.prod(profit), GRB.MINIMIZE)
f.optimize()
```

We changed the sense of the optimization to check if the model has a feasible solution.

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros
Coefficient statistics:
  Matrix range      [5e+00, 2e+01]
  Objective range   [5e+01, 8e+01]
  Bounds range      [0e+00, 0e+00]
  RHS range         [4e+02, 5e+02]
Presolve time: 0.01s
Presolved: 2 rows, 2 columns, 4 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   1.062500e+02   0.000000e+00      0s
       2    2.2000000e+03   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.02 seconds
Optimal objective  2.200000000e+03
```

```
# display optimal values of decision variables
if f.status == GRB.Status.OPTIMAL:
    print('Optimal solution found')
    print('total profits', f.objVal)
    for v in f.getVars():
        print(v.varName, v.x)
```

```
Optimal solution found
('total profits', 2200.0)
('make[table]', 14.0)
('make[chair]', 24.0)
```

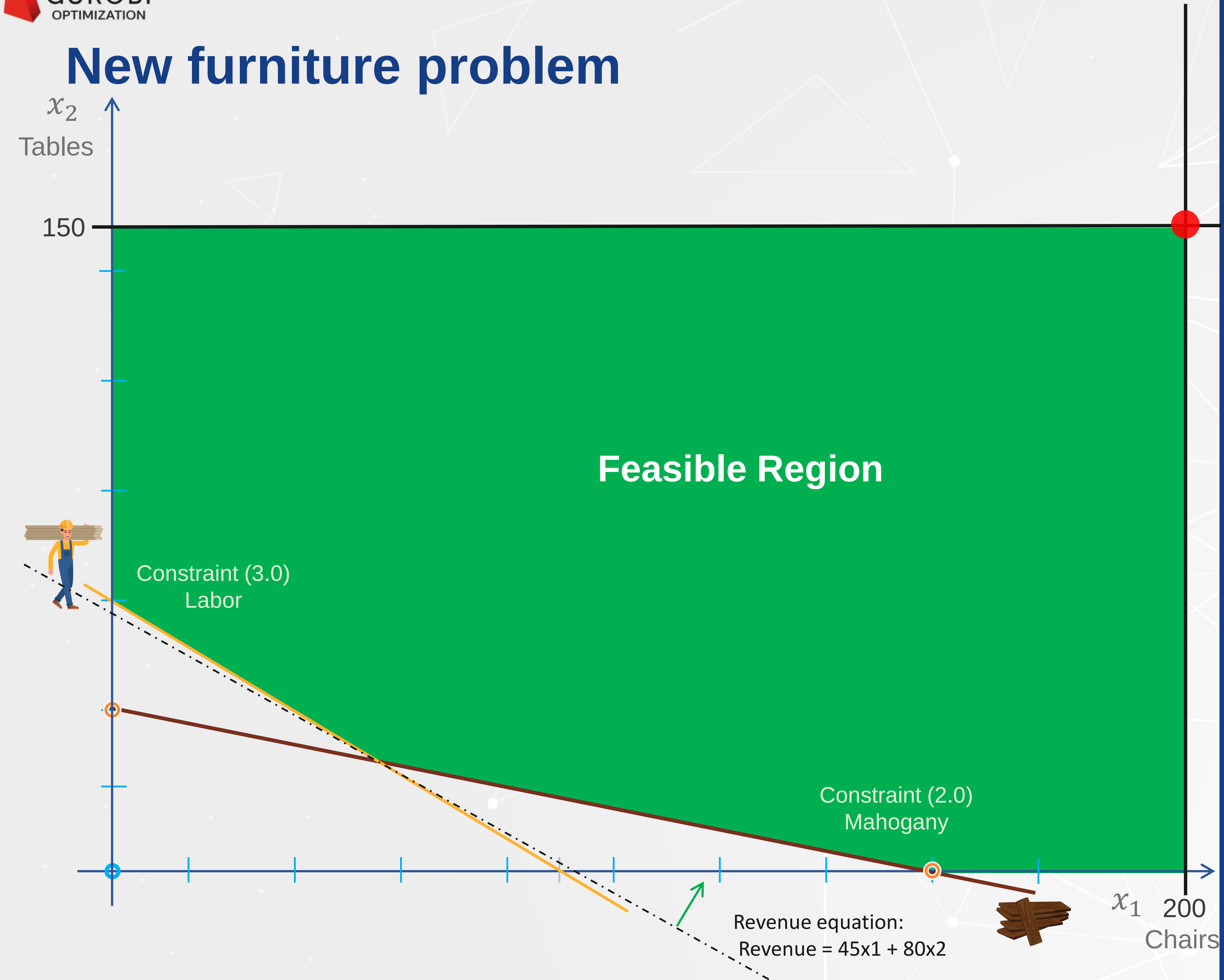
The Gurobi solver now finds a minimum optimal solution. Hence, the LP problem is feasible and for the revenue maximization problem, it is unbounded.

GUROBI OPTIMIZATION

New business conditions

- The data scientist recognized that it is unrealistic to assume that an infinite amount of chairs and tables can be sold.
- The data scientist using predictive analytics techniques has determined that at most 200 chairs and 150 tables can be sold.

New furniture problem



New business conditions

$(1.0)\quad Max\ revenue = 45x_1 + 80x_2$

$(2.0)\quad 5x_1 + 20x_2 \geq 400$ Mahogany constraint

$(3.0)\quad 10x_1 + 15x_2 \geq 450$ Labor constraint

$0 \leq x_1 \leq 200 \quad 0 \leq x_2 \leq 150$

Unbounded Furniture Problem: solved with Gurobi ..4

Let $price_p$ be the price of product $p \in products = \{chairs, \ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resources = \{mahogany, \ labor\}$.

Let's assume that we have an unlimited availability of resources. The only constraint is that the suppliers of mahogany and labor requires to consume at least 400 units of mahogany and 450 units of labor per planning period.

However, the marketing depatment has established that at most 200 chairs and 150 tables can be sold.

Let's define the upper bound on chairs and tables be the vector $upBound_p$

Let $bom_{r,j}$ be the amount of resource r required by product p. Then the general formulation of the Furniture problem is:

$$Max \sum_{p \in products} price_p make_p$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p \geq capacity_r \quad \forall\, r \in resources$$

$$0 \leq make_p \leq upBound_p \quad \forall\, p \in products$$

```
# products data,
products,  price, upBound = multidict({
  'chair': [45, 200],
  'table':  [80, 150] })
```

Adding the upper bounds to the multidict for products

Unbounded Furniture Problem: solved with Gurobi .. 5

```
# Declare and initialize model
f = Model('Furniture')
```

```
# Create decision variables for the products to make
make = f.addVars(products, ub=upBound, name="make")
```

Adding the upper bounds to the multidict for products

```
# Create an object of type list to store
#the constraints for each resource

res = f.addConstrs(((sum(bom[r,p]*make[p] for p in products) >= capacity[r]) for r in resources), name='R')
```

```
# The objective is to maximize total profit
f.setObjective(make.prod(profit), GRB.MAXIMIZE)
```

Unbounded Furniture Problem: solved with Gurobi .. 6

```
# save LP model for inspection
f.write('Furnture0065.lp')
```

```
\ Model Furniture
\ LP format - for model browsing. Use MPS format to capture full model detail.
Maximize
  80 make[table] + 45 make[chair]
Subject To
 R[mahogany]: 20 make[table] + 5 make[chair] >= 400
 R[labor]: 15 make[table] + 10 make[chair] >= 450
Bounds
 make[table] <= 150
 make[chair] <= 200
End
```

Unbounded Furniture Problem: solved with Gurobi .. 7

```
# run optimization engine
f.optimize()
```

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 2e+01]
  Objective range  [5e+01, 8e+01]
  Bounds range     [2e+02, 2e+02]
  RHS range        [4e+02, 5e+02]
Presolve removed 2 rows and 2 columns
Presolve time: 0.04s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    2.1000000e+04   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.15 seconds
Optimal objective  2.100000000e+04
```

```
# display optimal values of decision variables
if f.status == GRB.Status.OPTIMAL:
    print('Optimal solution found')
    print('total revenue', f.objVal)
    for v in f.getVars():
        print(v.varName, v.x)
```

```
Optimal solution found
('total revenue', 21000.0)
('make[table]', 150.0)
('make[chair]', 200.0)
```

As expected, Gurobi solver now finds an optimal solution.

The optimal number of chairs to make is equal to the chairs upper bound and the optimal number of tables to make is equal to the tables upper bound.

The optimal objective function value is a total revenue of $21,000

Infeasible LP problem

Gurobi Python API

Minimum Optimal Solution

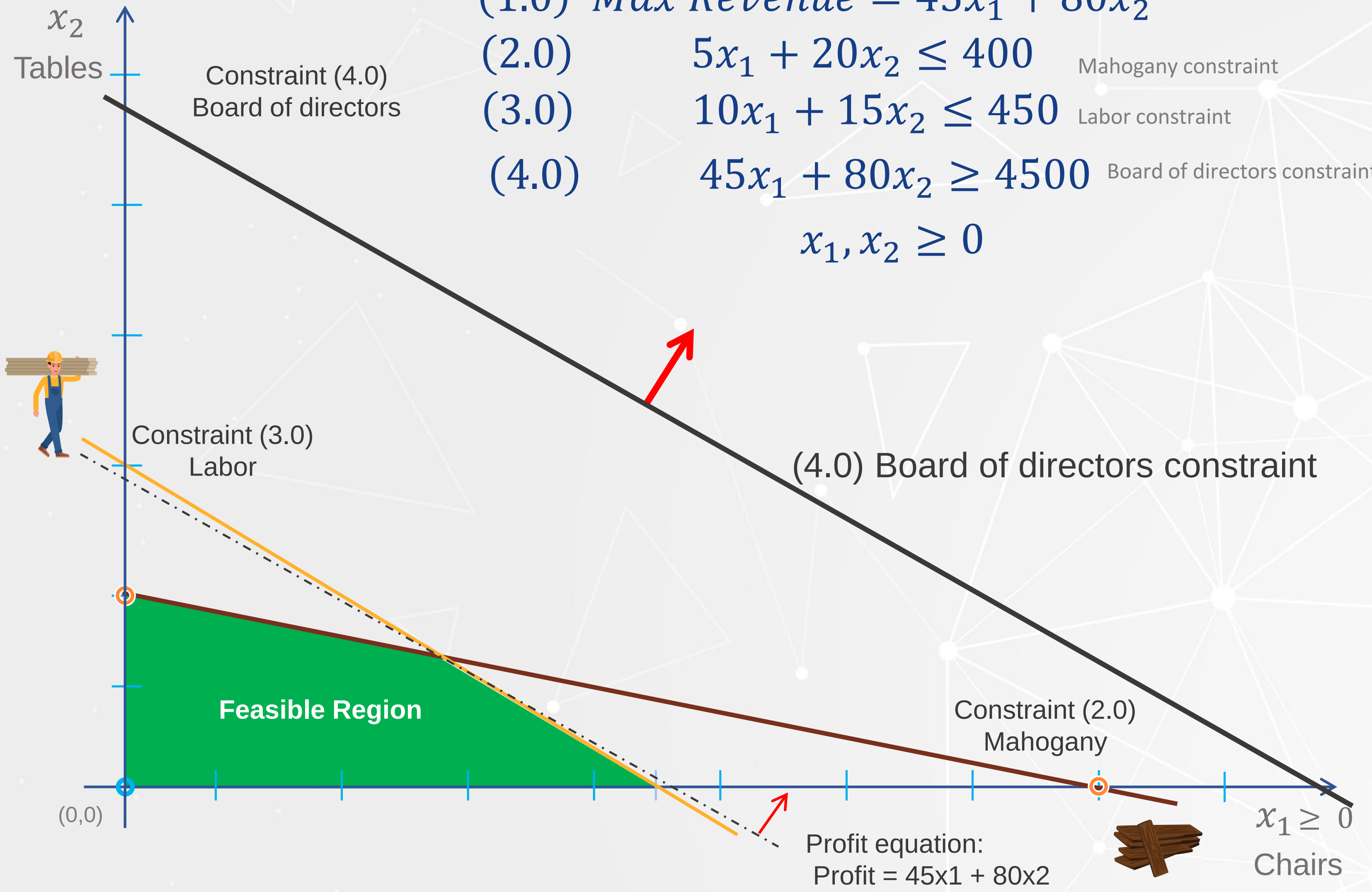
New business conditions

- The data scientist receives a memo from the CEO of the furniture company saying that the new board of directors of the company requires a total revenue of at least $4,500 per week.

Furniture Problem

The new LP problem formulations is:

$$(1.0)\quad Max\ Revenue = 45x_1 + 80x_2$$
$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Mahogany constraint}$$
$$(3.0)\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor constraint}$$
$$(4.0)\quad 45x_1 + 80x_2 \geq 4500 \quad \text{Board of directors constraint}$$
$$x_1, x_2 \geq 0$$



New business conditions

- Evidently, there are no points that satisfy all the constraints simultaneously. Hence this LP problem is infeasible.

Infeasible Furniture Problem: using with Gurobi .. 1

Let $price_p$ be the price of product $p \in products = \{chairs,\ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resources = \{mahogany,\ labor\}$.

The board of directors has imposed a constraint of a minimum revenue, $minRev = \$4,500$, per week.

Let $bom_{r,j}$ be the amount of resource r required by product p. Then the general formulation of the Furniture problem is:

$$Max \sum_{p \in products} price_p make_p$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p \leq capacity_r \quad \forall\, r \in resources$$

$$\sum_{p \in products} price_p make_p \geq minRev$$

$$make_p \geq 0 \quad \forall\, p \in products$$

```python
minRev = 4500
```

Defining a new parameter to capture the minimum revenue value that the board of directors may impose.

```python
# Board constraint of minimum revenue
minProfitConstr = f.addConstr((sum(price[p]*make[p] for p in products) >= minRev), name='B')
```

Adding a new constraint to the f model to ensure that the minimum revenue impose by the Board is satisfied.

Infeasible Furniture Problem: solved with Gurobi .. 2

```python
# save model for inspection
f.write('furnitureB.lp')
```

```
\ Model Furniture
\ LP format - for model browsing. Use MPS format to capture full model detail.
Maximize
  80 make[table] + 45 make[chair]
Subject To
 R[mahogany]: 20 make[table] + 5 make[chair] <= 400
 R[labor]: 15 make[table] + 10 make[chair] <= 450
 B: 80 make[table] + 45 make[chair] >= 4500
Bounds
End
```

Infeasible Furniture Problem: solved with Gurobi .. 3

```
# run optimization engine
f.optimize()
```

```
Optimize a model with 3 rows, 2 columns and 6 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 8e+01]
  Objective range  [5e+01, 8e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+03]
Presolve time: 0.03s

Solved in 0 iterations and 0.03 seconds
Infeasible or unbounded model
```

```
# display optimal values of decision variables
if f.status == GRB.Status.OPTIMAL:
    print('Optimal solution found')
    print('total revenue', f.objVal)
    for v in f.getVars():
        print(v.varName, v.x)
```

We check the status of the f model to see if Gurobi found an optimal solution. If true, then we print the optimal solution and the associated objective function value.

Infeasible Furniture Problem: solved with Gurobi .. 3

```
#Check if model infeasible or unbounded
if f.status == GRB.Status.INF_OR_UNBD :
    print('LP problem is either infeasible or unbounded')
    print('Checking if LP problem is feasible')
    print('set objective function to zero value and re-run engine' )
    f.setObjective( 0, GRB.MAXIMIZE)
    f.optimize()
```

```
LP problem is either infeasible or unbounded
Checking if LP problem is feasible
set objective function to zero value and re-run engine
Optimize a model with 3 rows, 2 columns and 6 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 8e+01]
  Objective range  [0e+00, 0e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+03]
Presolve time: 0.01s

Solved in 0 iterations and 0.01 seconds
Infeasible model
```

We set an objective with zero value. Then run the Gurobi solver to find any feasible solution.

```
#Check if model with zero objective is infeasible
if f.status == GRB.Status.INFEASIBLE :
    print('LP problem is proven to be infeasible')
```

```
LP problem is proven to be infeasible
```

We check the status of the f model to see if it is infeasible. If true, then we print that we have proven that the model is infeasible.

Infeasible Furniture Problem

Addressing infeasibility

- There is a Board of Director meeting and the data scientist cannot tell the Board that their requirement of having a total revenue of at least $4,500 per week is Infeasible !!!
- The data scientist knows that the cause of infeasibility is the limited capacity available of resources that does not allow the production of chairs and tables to reach the minimum level of total revenue.
- The data scientist calls the supplier of mahogany and talks with the labor union and they agree to increase the supply at a cost. An extra unit of mahogany will cost $20 and one hour of overtime of labor will cost $30.

Furniture Problem: addressing infeasibility .. 2

Furniture model formulation: buying extra supply

Let $price_p$ be the price of product $p \in products = \{chairs,\ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resources = \{mahogany,\ labor\}$.

The board of directors has imposed a constraint of a minimum revenue, $minRev = \$4,500$, per week.

To satisfy Board of Directors demands, the supplier of mahogany can get extra mahogany at a rate of \$20 per unit; and the labor union can provide overtime of labor at a rate of \$30 per unit.

Let's define a new decision variable, $extra_r$, that measures the extra capacity of resource $r \in resources = \{mahogany,\ labor\}$ which are required to meet the Board of Directors demands. Let the cost of extra capacity be defined by the parameter $extraCost_r$ for each resource $r \in resources = \{mahogany,\ labor\}$.

Let $bom_{r,j}$ be the amount of resource r required by product p. Then the general formulation of the Furniture problem is:

$$Max \sum_{p \in products} price_p make_p - \sum_{r \in resources} extraCost_r extra_r$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p - extra_r \leq capacity_r \quad \forall\ r \in resources$$

$$\sum_{p \in products} price_p make_p \geq minRev$$

$$make_p \geq 0 \quad \forall\ p \in products$$

$$extra_r \geq 0 \quad \forall\ r \in resoruces$$

Furniture Problem: addressing infeasibility .. 2

$$\text{Max} \sum_{p \in products} price_p make_p - \sum_{r \in resources} extraCost_r extra_r$$

Subject to :

$$\sum_{p \in products} bom_{r,p} make_p - extra_r \le capacity_r \quad \forall\, r \in resources$$

$$\sum_{p \in products} price_p make_p \ge minRev$$

$$make_p \ge 0 \quad \forall\, p \in products$$

$$extra_r \ge 0 \quad \forall\, r \in resorces$$

```
# resources data
resources, capacity, extraCost = multidict({
  'mahogany': [400, 20],
  'labor':  [450, 30] })
```

We include in the resources multidict the extra cost of adding resource capacity.

```
# Create decision variables for extra capacity of resources
extra = f.addVars(resources, name="extra")
```

Create a new decision variable to measure the extra resource capacity to meet the Board constraint.

```
# Create an object of type list to store
#the constraints for each resource
```

$$\sum_{p \in products} bom_{r,p} make_p - extra_r \le capacity_r \quad \forall\, r \in resources$$

```
res = f.addConstrs(((sum(bom[r,p]*make[p] for p in products) - extra[r] <= capacity[r]) for r in resources), name='R')
```

We add a decision variable (extra) to the RHS of each resource constraint to add the extra capacity to satisfy the Board constraint.

```
# The objective is to maximize total revenue
f.setObjective(make.prod(price) - extra.prod(extraCost), GRB.MAXIMIZE)
```

$$\text{Max} \sum_{p \in products} price_p make_p - \sum_{r \in resources} extraCost_r extra_r$$

We subtract the cost of the extra capacity to meet the Board constraint .

Furniture Problem: addressing infeasibility .. 3

```
# save model for inspection
f.write('furnitureB.lp')
```

```
\ Model Furniture
\ LP format - for model browsing. Use MPS format to capture full model detail.
Maximize
  80 make[table] + 45 make[chair] - 20 extra[mahogany] - 30 extra[labor]
Subject To
 R[mahogany]: 20 make[table] + 5 make[chair] - extra[mahogany] <= 400
 R[labor]: 15 make[table] + 10 make[chair] - extra[labor] <= 450
 B: 80 make[table] + 45 make[chair] >= 4500
Bounds
End
```

Furniture Problem: addressing infeasibility .. 3

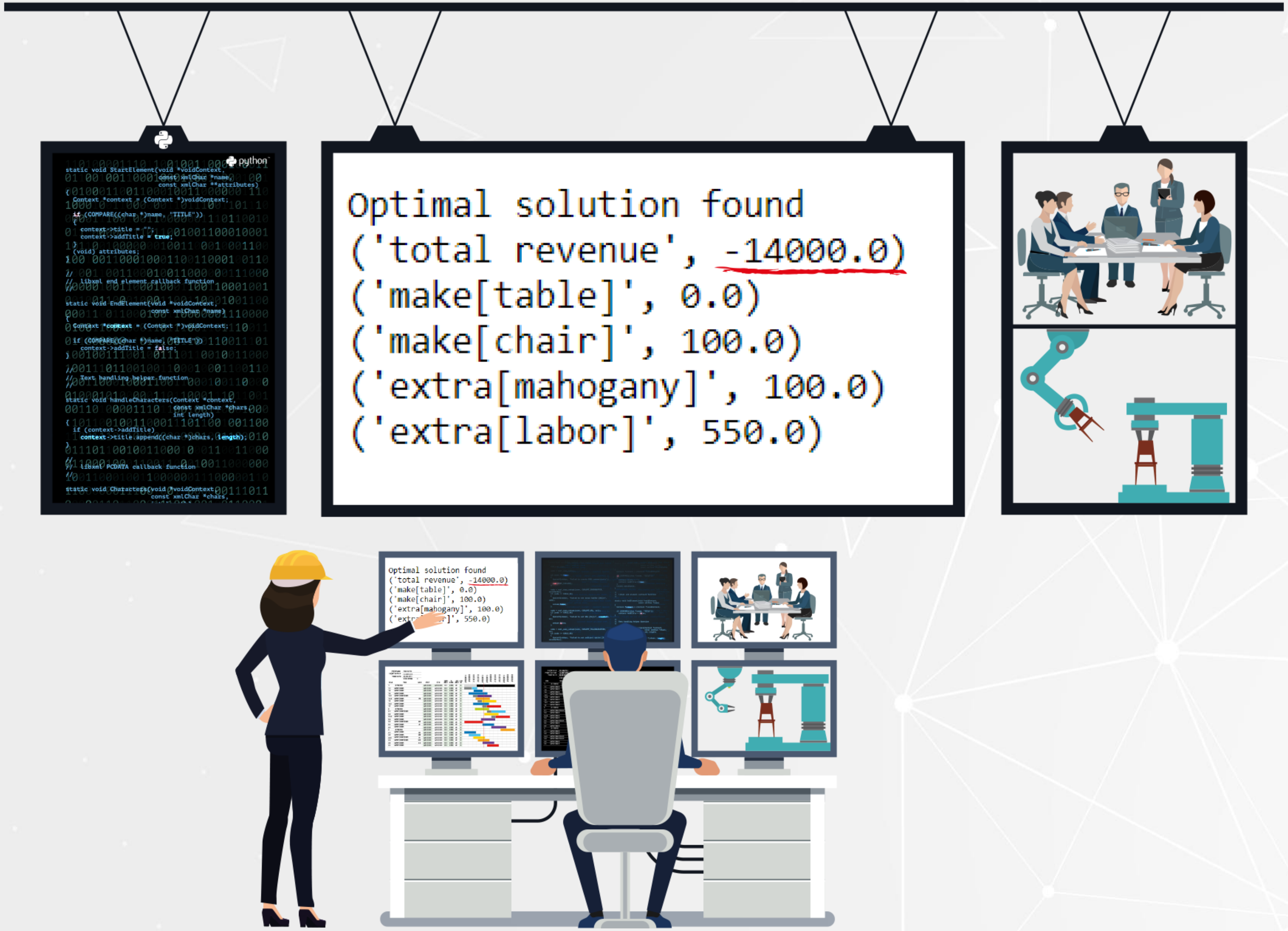
```
# run optimization engine
f.optimize()
```

```
Optimize a model with 3 rows, 4 columns and 8 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 8e+01]
  Objective range  [2e+01, 8e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+03]
Presolve time: 0.10s
Presolved: 3 rows, 4 columns, 8 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.0625000e+31   1.015625e+30   1.062500e+01      0s
       4   -1.4000000e+04   0.000000e+00   0.000000e+00      0s

Solved in 4 iterations and 0.16 seconds
Optimal objective -1.400000000e+04
```

```
# display optimal values of decision variables
if f.status == GRB.Status.OPTIMAL:
    print('Optimal solution found')
    print('total revenue', f.objVal)
    for v in f.getVars():
        print(v.varName, v.x)
```

Gurobi solver found an optimal solution.

Furniture Problem: addressing infeasibility



To meet the Board request requires 100 extra units of mahogany and 550 extra units of overtime.
The optimal production plan is to produce only 100 chairs.

The company will lose $14,000 per week !!!

The Board request was proven not to be a good idea.

GUROBI
OPTIMIZATION

Final Remarks about model status

- It is a good modeling practice that the LP problem is well characterized.
  - This means that when the data of the LP problem satisfy the specified modeling assumptions, the LP problem has an optimal solution.
  - That is, the LP problem is never infeasible, unbounded, and there are no alternate optimal solutions.
    - To avoid infeasibility add “artificial” variables to the constraints that may be infeasible. These artificial variables will have a “high” penalty in the objective function in such a way that they become positive only to make the problem feasible. Ideally, these artificial variables have a business meaning that the user of the decision support application can properly interpret.
    - To avoid that the LP problem is unbounded, define realistic upper bounds for all the decision variables of the LP problem.
    - If the LP problem has alternate optimal solutions, if possible add another objective functions that eliminates the alternate optimal solutions.

Linear Programming Overview

Further considerations:

- Maximize or minimize objective function
- Unconstrained decision variables
- Initial basic solution
- Presolve
- Matrix sparsity

What if we need to minimize an objective function?

We have assumed that we are maximizing an LP problem. Notice that the rules for a non-basic variables to become basic variables can be changed to address the minimization problem.

Alternatively, since $\boldsymbol{Min}\ \sum_{j=1}^{n} \boldsymbol{b_j x_j}\ = -\boldsymbol{Max}\ -\sum_{j=1}^{n} \boldsymbol{b_j x_j}$, then we solve $\boldsymbol{Max}\ \sum_{j=1}^{n} (-\boldsymbol{b_j})\boldsymbol{x_j}$.

Example: Min {1, 2} = -Max {-1, -2} = 1

```
# The objective is to maximize total revenue
f.setObjective(make.prod(price) - extra.prod(extraCost), GRB.MAXIMIZE)
```

Note: The default sense in Gurobi is to minimize the objective function of the LP problem we want to solve. If we want to maximize the objective we need to use the Gurobi argument (GRB.MAXIMIZE) when defining the objective function.

GUROBI OPTIMIZATION

What if a decision variable can be positive or negative?

If the decision variable is unconstrained in sign, the Gurobi Optimizer will take care of this automatically.

An example of a decision variable that is unconstrained is profit. Negative profit is interpreted as a loss.

How to determine an initial basic feasible solution?

- **Gurobi behind the scenes may use a technique called Phase 1 of Linear Programming:**
  - This technique entails adding non-negative artificial variables to the equations (=) and greater-or-equal (≥) inequalities.
  - Then replace the original objective function by a new objective, minimize the summation of the artificial variables.
  - If the minimum value of the summation of artificial variables is positive, the LP problem is infeasible.
  - If the minimum value of the summation of artificial variables is zero, then the basic feasible variables of the optimal solution are an initial basic feasible solution of the original LP model (*).

(*) There are some subtleties about this statement beyond the scope of this class.

How to determine an initial basic feasible solution? .. 2

- Consider the following LP problem with the Furniture factory board request.

$$Max \sum_{p \in products} price_p make_p - \sum_{r \in resources} extraCost_r extra_r$$

$$Subject\ to:$$

$$\sum_{p \in products} bom_{r,p} make_p - extra_r \leq capacity_r \quad \forall\, r \in resources$$

$$\sum_{p \in products} price_p make_p \geq minRev$$

$$make_p \geq 0 \quad \forall\, p \in products$$

$$extra_r \geq 0 \quad \forall\, r \in resoruces$$

- An instance of the parametrized formulation is

$$Max\ z = 45x_1 + 80x_2 \quad - 20x_3 \ - 30x_4$$

$$5x_1 + 20x_2 - x_3 \leq 400$$

$$10x_1 + 15x_2 - x_4 \leq 450$$

$$45x_1 + 80x_2 \geq 4500$$

$$x_1, x_{2,}, x_3. x_4 \geq 0$$

How to determine an initial basic feasible solution? .. 2

- Original LP problem.

$$Max\ z = 45x_1 + 80x_2 \quad - 20x_3 \ - 30x_4$$
$$5x_1 + 20x_2 - x_3 \le 400$$
$$10x_1 + 15x_2 - x_4 \le 450$$
$$45x_1 + 80x_2 \ge 4500$$
$$x_1, x_{2,}, x_3 . x_4 \ge 0$$

- We add an artificial variable alpha to the ≥ constraint. At Phase 1 of linear programming, we minimize the value of alpha, i.e. alpha should be equal to zero and non-basic variable, to identify an initial basic feasible solution.

Original LP Problem

$$Min\ w = \alpha$$
$$5x_1 + 20x_2 - x_3 \le 400$$
$$10x_1 + 15x_2 - x_4 \le 450$$
$$45x_1 + 80x_2 + \alpha \ge 4500$$
$$x_1, x_2, x_3 . x_4, \alpha \ge 0$$

LP Problem Standard Form

$$Max\ w = -\alpha$$
$$5x_1 + 20x_2 - x_3 + h_1 = 400$$
$$10x_1 + 15x_2 - x_4 + \ h_2 = 450$$
$$45x_1 + 80x_2 + \alpha - s_1 = 4500$$
$$x_1, x_2, x_3 . x_4, \alpha, h_1, h_2, s_1 \ge 0$$

LP Problem Canonical Form

$$Max\ w = -4500 + 45x_1 + 80x_2 - s_1$$
$$h_1 = 400 \ - 5x_1 - 20x_2 + x_3$$
$$h_2 = 450 \ - 10x_1 - 15x_2 + x_4$$
$$\alpha = 4500 \ - 45x_1 - 80x_2 + s_1$$
$$x_1, x_2, x_3 . x_4, \alpha, h_1, h_2, s_1 \ge 0$$

How to determine an initial basic feasible solution? .. 3

We apply the simplex method to the Phase 1 LP problem in canonical form with respect to the basis (h_1, h_2, α). The following table shows the basic feasible solution at each iteration of the simplex method, the non basic variable entering the basis and the basic variable leaving the basis.

Iteration	Basic feasible solution Phase 1	Non basic variable entering the basis	Basic variable leaving the basis
1	$h_1 = 400$, $h_2 = 450$, $\alpha = 4500$	x_2	h_1
2	$x_2 = 20$, $h_2 = 150$, $\alpha = 2900$	x_3	h_2
3	$x_2 = 30$, $x_3 = 200$, $\alpha = 2100$	x_4	α
4	$x_2 = 56.25$, $x_3 = 725$, $x_4 = 393.75$	x_1	x_2
5	$x_1 = 100$, $x_3 = 100$, $x_4 = 550$	This solution is optimal.	

Notice that α is non basic and = 0. This solution is an initial basic feasible solution of the original LP problem.

Iteration 4 continues with Phase 2 of the simplex method, where we find the optimal feasible basic solution.

GUROBI OPTIMIZATION

Optimal solution found using Gurobi

```
# run optimization engine
f.optimize()
```

```
Optimize a model with 3 rows, 4 columns and 8 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 8e+01]
  Objective range  [2e+01, 8e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [4e+02, 5e+03]
Presolve time: 0.10s
Presolved: 3 rows, 4 columns, 8 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.0625000e+31   1.015625e+30   1.062500e+01      0s
       4   -1.4000000e+04   0.000000e+00   0.000000e+00      0s

Solved in 4 iterations and 0.16 seconds
Optimal objective -1.400000000e+04
```

```
# display optimal values of decision variables
if f.status == GRB.Status.OPTIMAL:
    print('Optimal solution found')
    print('total revenue', f.objVal)
    for v in f.getVars():
        print(v.varName, v.x)
```

```
Optimal solution found
('total revenue', -14000.0)
('make[table]', 0.0)
('make[chair]', 100.0)          → x1 = 100
('extra[mahogany]', 100.0)      → x3 = 100
('extra[labor]', 550.0)         → x4 = 550
```

Gurobi corroborates the optimal solution found manually using the two phases simplex method approach

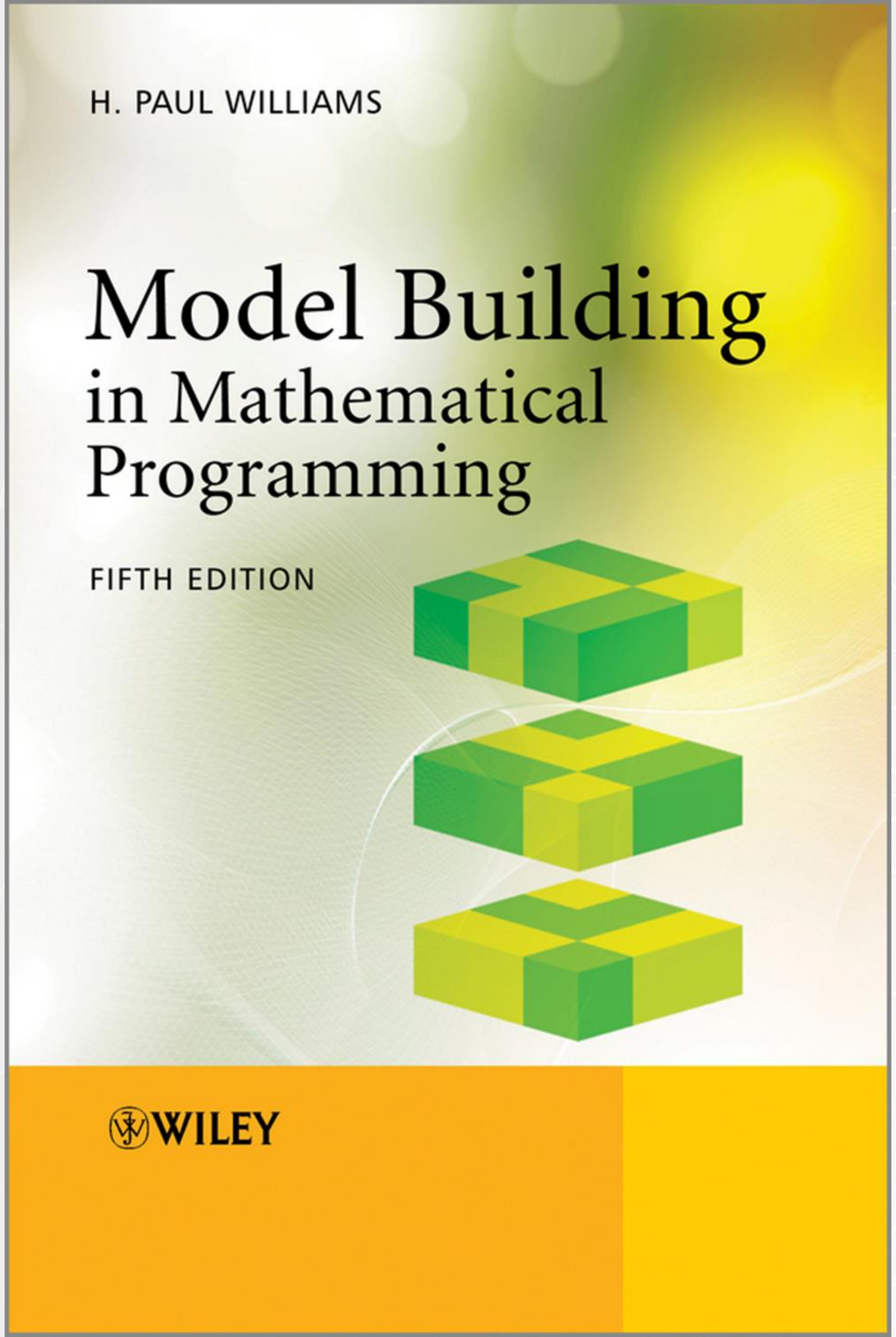
How to determine an initial basic feasible solution? .. 4

Remarks

- Gurobi user does not need to provide an initial basic feasible solution, or solve the phase 1 minimization of the sum of artificial variables problem to generate an initial basic solution.

- Gurobi only needs the original LP formulation. It automatically finds an initial feasible basic solution to start the simplex method.

Presolve


H. PAUL WILLIAMS
Model Building
in Mathematical
Programming
FIFTH EDITION
WILEY

Presolve

- LP problems can use a large amount of computer time, consequently it is advisable to have LP models that can be solved as quickly as possible.
- The Presolve engine of Gurobi can dramatically reduce the size of an LP problem. The reduced problem can then be solved faster than the original one. The solution of the reduced problem is then used to generate a solution of the original problem.
- To briefly illustrate the core ideas behind a Presolve approach consider the following example.

- Max $2x1 + 3x2 - x3 - x4$ … (0)
- Subject to: $x1 + x2 + x3 - 2x4 \leq 4$ … (1)
- $-x1 - x2 + x3 - x4 \leq 1$ … (2)
- $x1 + x4 \leq 1$ … (3)
- $x1, x2, x3, x4 \geq 0$ … (4)

- Notice that x3 has a negative objective coefficient. We have a maximization problem, then we want to make x3 as small as possible.
- Observe that x3 has positive coefficient in constraints (1) and (2),
- and these constraints are of the '≤', then we want to make x3 as small as possible.
- Therefore, x3 can be reduced to its lower bound of zero and eliminated as a redundant variable.

Presolve ..2

- Max $2x1 + 3x2 - x4$ … (0)
- Subject to: $x1 + x2 - 2x4 \leq 4$ … (1)
- $-x1 - x2 - x4 \leq 1$ … (2)
- $x1 + x4 \leq 1$ … (3)
- $x1, x2, x3, x4 \geq 0$ … (4)

- After removing x3, let's analyze constraint (2) $-x1 - x2 - x4 \leq 1$. Notice that all coefficients of the variables in this constraint are negative. For any positive value of variables x1, x2, and x3 constraint (2) is satisfied, consequently this constrained is redundant and can be eliminated. The reduced problem is then:

This simple Presolve approach eliminated one variable and one constraint.

- Max $2x1 + 3x2 - x4$ … (0)
- Subject to: $x1 + x2 - 2x4 \leq 4$ … (1)
- $x1 + x4 \leq 1$ … (3)
- $x1, x2, x4 \geq 0$ and $x3=0$ … (4)

With large models, a Presolve approach could lead to significant reductions in the amount of computation needed to solve the model. I have seen Gurobi Presolve reduce a large scale problem to less than 10% of its original size.

Example of the power of Gurobi Presolve

Gurobi Case Study:

- **Solution approach:**
- **Problem statement:** how to optimize the selection and scheduling of a portfolio of IT projects such that the trade-offs among various objectives are optimized, while satisfying resource constraints (e.g. labor availability and budgets) and other business constraints (e.g. project precedence constraints).
- A decision support tool, called Project Portfolio Optimization (PPO), was built to automate the number crunching processes during the creation and maintenance of a portfolio of IT projects.
- Gurobi solver was used to solve this complex MIP problem.

Example of the power of Gurobi Presolve .. 3

Size	PPO model	PPO model after Presolve	Reduction %
Total # of variables	9307	3170	66 %
Total # of constraints	6461	337	94 %
Gurobi solving time	–	8.53 s	–
Gurobi solving time no presolve	156.1 s	–	

Gurobi with Presolve runs 17X faster

Matrix sparsity of LP problems

- Presolve takes advantage of the sparsity of the matrix of technology coefficients to significantly reduce the size of the problem.
- This means that a small number of coefficients in the matrix of technology coefficients is non-zero.

- An important feature of practical LP and MIP models is that their matrix of technology coefficients is sparse.
- The sparsity of the matrix of technology coefficients is effectively exploited in a very efficient computer implementation of the simplex method called the **revised simplex method**.

Objective function coefficients

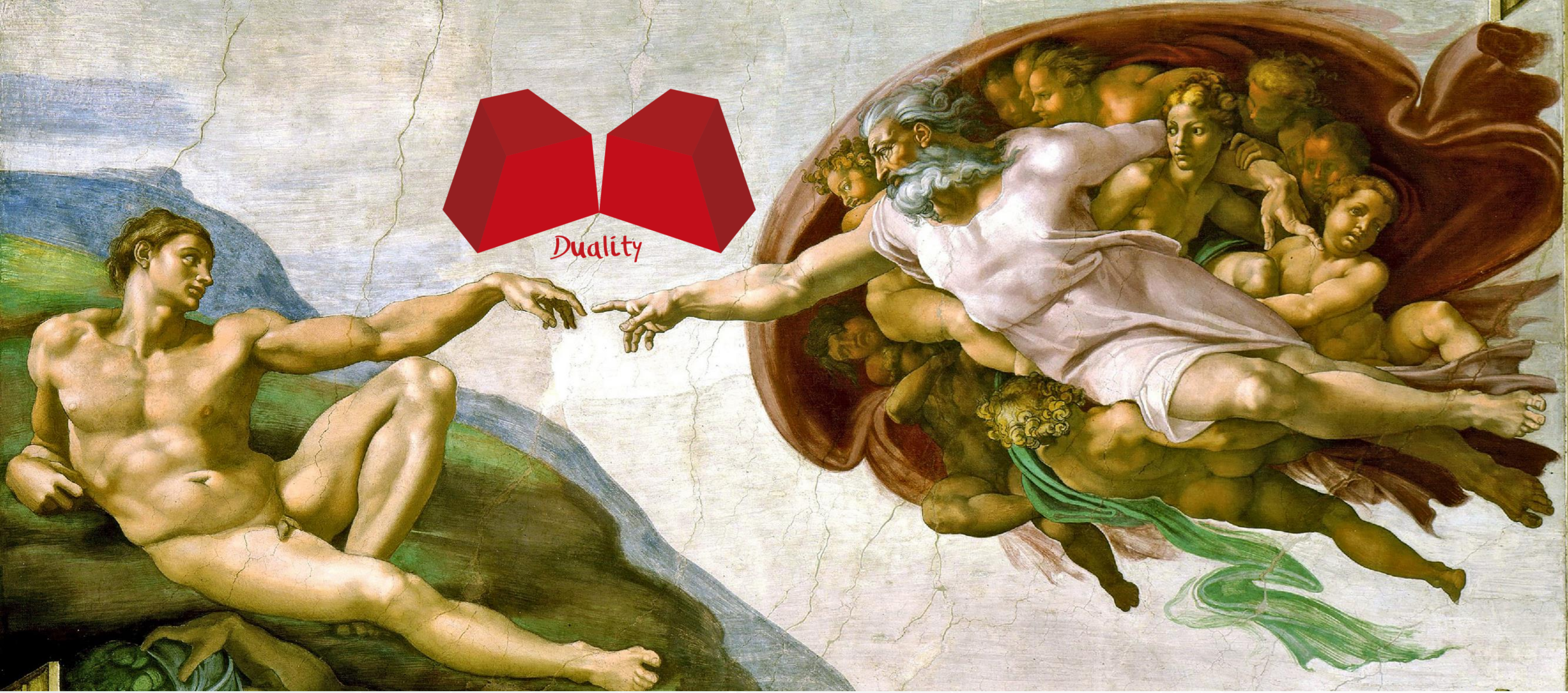
Objective Function → $$Max \sum_{j=1}^{n} b_j x_j$$ → Decision Variables

RHS: right hand side

Constraints → $$\sum_{j=1}^{n} a_{i,j} x_j \leq K_i \ (i = 1..m)$$ → Set of indices for Constraints

Technology coefficients

$$x_j \geq 0 \ (j = 1..n)$$ → Set of indices for Variables



Duality in Linear Programming

Linear Programming Duality

In the section of sensitivity analysis of Linear Programming, we asked the following question:

- For the furniture problem, is it profitable to make a third product, like desks?
  - Assume that the price of the desk is $110,
  - and the desk consumes 15 units of mahogany and 25 units of labor.
- Shadow prices determine the marginal worth of an additional unit of a resource:
  - The shadow price of the mahogany capacity constraint is $1.
  - The shadow price of the labor capacity constraint is $4.
- Let's compute the opportunity cost of making one desk and compare it with the price of a desk. If this opportunity cost is greater than the desk price, then it is not worth it to make desks.
  - The opportunity cost can be computed by multiplying the units of mahogany capacity that one desk built consumes by the shadow price of mahogany capacity, and multiplying the units of labor capacity that one desk built consumes by the shadow price of labor capacity:
    - That is, ($1)*15 (units of mahogany) + ($4)*25 (hours of labor) = $115 > $110.
- Therefore, investing resources to produce desks, otherwise used to produce chairs and tables, is not profitable.

Linear Programming Duality .. 2

Duality in Linear Programming is an unifying theory that established the relation between an LP problem – called **Primal Problem**, and another related LP problem –called **Dual Problem,** where its decision variables (**dual variables**) are the shadow prices of the resource constraints.

Furniture Problem: Primal and Dual problems

The primal and dual of the Furniture problem are:

Primal

$$(1.0) \quad Max\ revenue = 45x_1 + 80x_2$$

$$(2.0) \quad 5x_1 + 20x_2 \leq 400 \quad \text{Mahogany}$$

$$(3.0) \quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor}$$

$$x_1, x_2 \geq 0$$

Chairs Tables

Dual

$$(4.0) \quad Min\ Cost = 400w_1 + 450w_2$$

$$(5.0) \quad 5w_1 + 10w_2 \geq 45 \quad \text{Chairs}$$

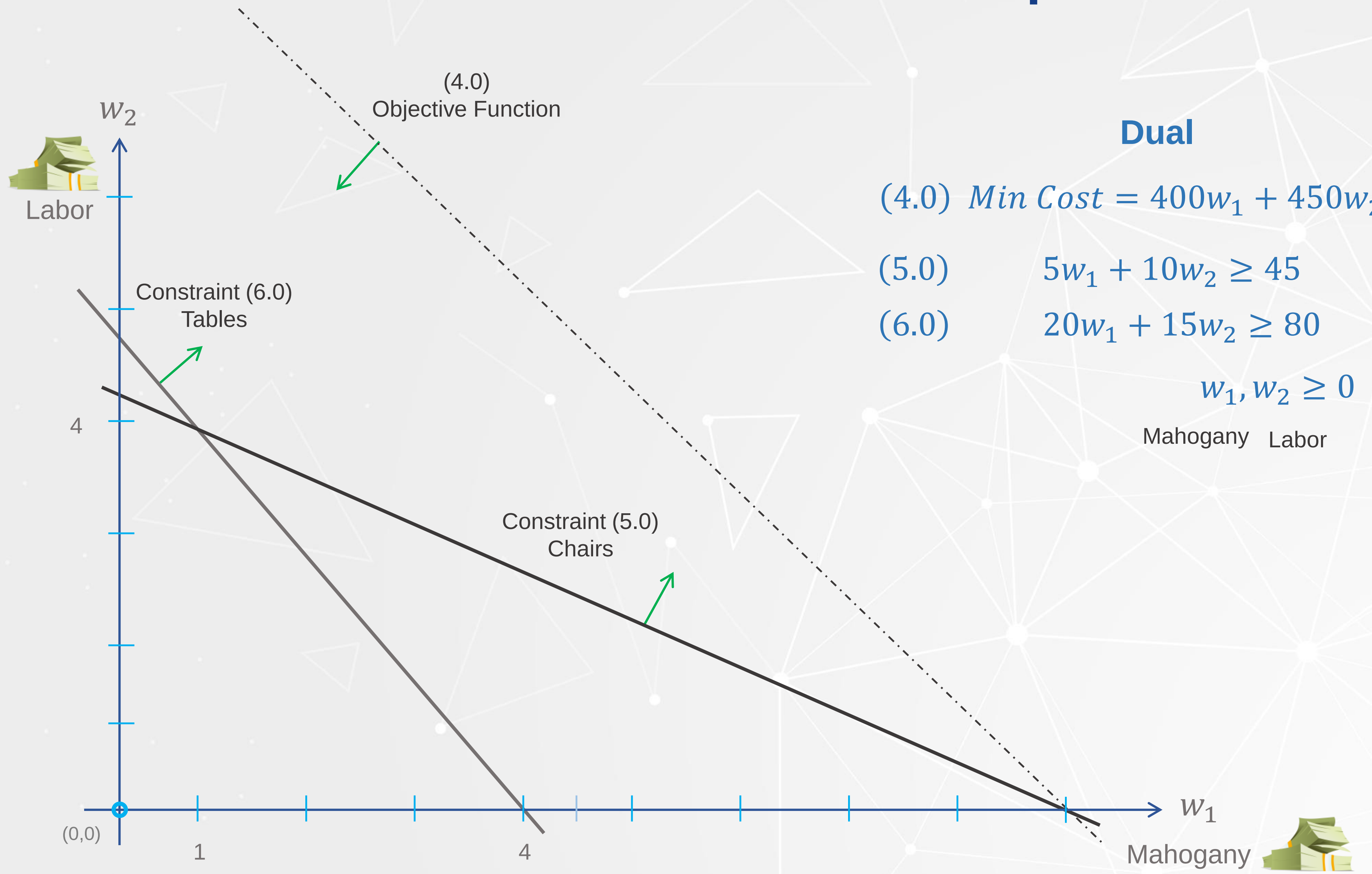
$$(6.0) \quad 20w_1 + 15w_2 \geq 80 \quad \text{Tables}$$

$$w_1, w_2 \geq 0$$

Mahogany Labor

- In this dual problem, the decision variable w1 represents the opportunity cost of the mahogany resource, and w2 is the opportunity cost of the labor resource. These decision variables are the shadow prices of mahogany and labor capacity.
- Notice the switch between the objective function coefficients and the right hand sides of the primal and dual problems.
- Also, notice that the rows of the primal problem are the columns of the dual problem. This means that inequalities (5.0) and (6.0) ensures that the opportunity costs of consumption of resources per unit of production of chairs and tables, respectively, should be at least the value of their price. The objective is to minimize the resource opportunity costs.

Furniture Dual Problem: Graphical solution



Dual

(4.0) $\quad Min\ Cost = 400w_1 + 450w_2$

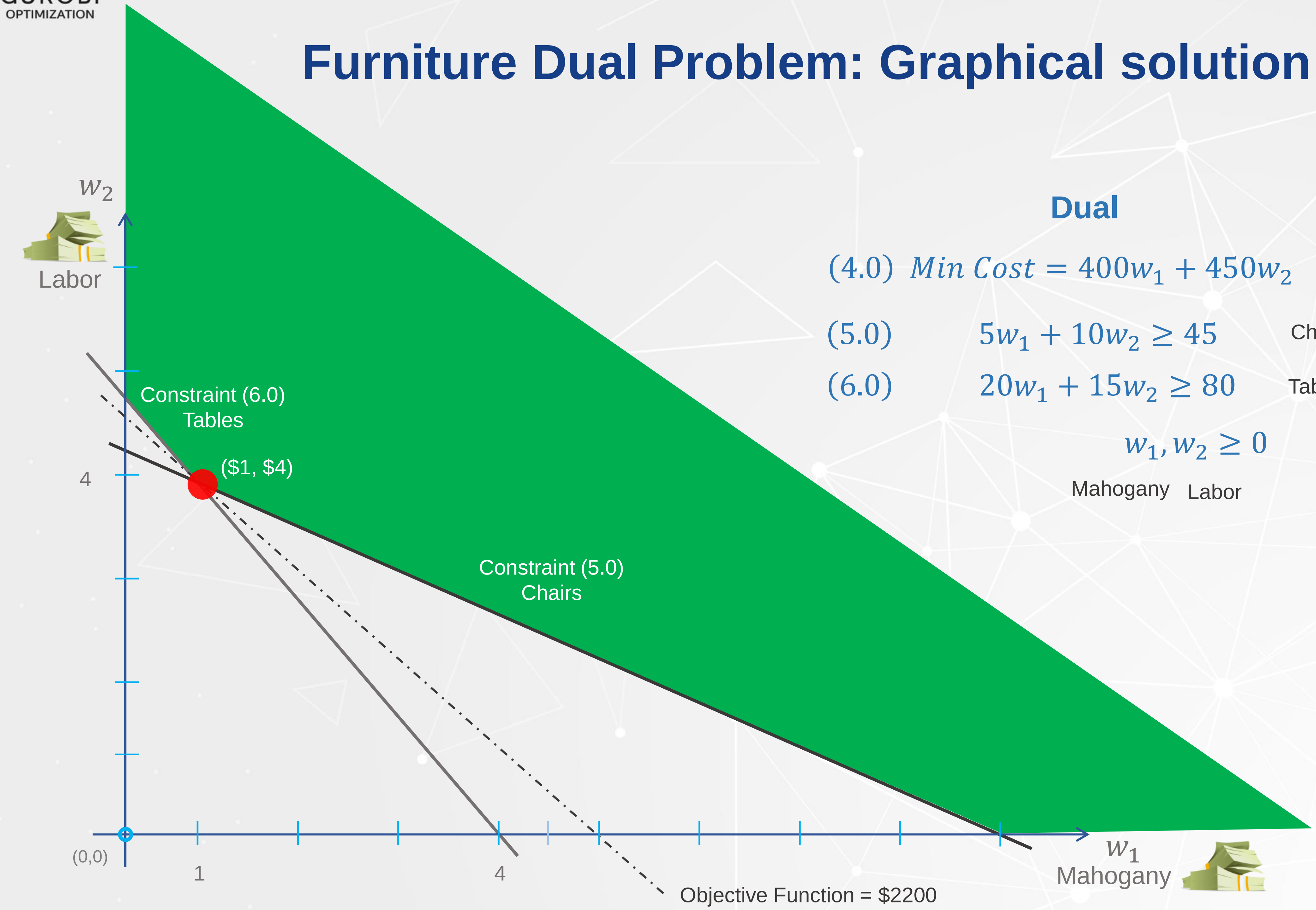
(5.0) $\quad 5w_1 + 10w_2 \geq 45$ Chairs

(6.0) $\quad 20w_1 + 15w_2 \geq 80$ Tables

$w_1, w_2 \geq 0$

Mahogany Labor

Furniture Dual Problem: Graphical solution



Dual

$$(4.0)\quad Min\ Cost = 400w_1 + 450w_2$$

$$(5.0)\quad 5w_1 + 10w_2 \geq 45 \quad \text{Chairs}$$

$$(6.0)\quad 20w_1 + 15w_2 \geq 80 \quad \text{Tables}$$

$$w_1, w_2 \geq 0$$

Mahogany Labor

Furniture Dual Problem: Parametrized

Furniture dual problem

Let $price_p$ be the price of product $p \in products = \{chairs, \ tables\}$, and let $capacity_r$ be the capacity available of resource $r \in resources = \{mahogany, \ labor\}$. Let $bom_{r,j}$ be the amount of resource r required by product p.

Let $shadowPrice_r$ be the shadow price, or opportunity cost, of resource $r \in resoruces = \{mahogany, \ labor\}$

$$Min \sum_{r \in resource} capacity_r shadowPrice_r$$

$$Subject \ to:$$

$$\sum_{r \in resource} bom_{r,p} shadowPrice_r \geq price_p \quad \forall \ p \in products$$

$$shadowPrice_r \geq 0 \quad \forall \ r \in resources$$

Furniture Dual Problem: Solved with Gurobi ..1

The data of the Furniture Dual problem **is identical** to the original Furniture problem –called Primal Problem.

```
# resources data
resources, capacity = multidict({
  'mahogany': 400,
  'labor':  450 })
```

```
# products data,
products, price = multidict({
  'chair': 45,
  'table':  80 })
```

```
# Bill of materials: resources required by each product
bom = {
  ('mahogany', 'chair'):  5,
  ('mahogany', 'table'): 20,
  ('labor',    'chair'): 10,
  ('labor',    'table'): 15 }
```

Furniture Dual Problem: Solved with Gurobi ..2

The right hand side of the constraints are the price of each product. The left hand side is the opportunity cost of consuming each resource when making the products. The sense of the inequalities is greater than equal to have an evaluation of the resource at least equal to the price.

```
# Declare and initialize model
f = Model('Furniture')
```

```
# Create decision variables for the resources capacity
shadowPrice = f.addVars(resources, name="price")
```

$$\sum_{r \in resource} bom_{r,p} shadowPrice_r \geq price_p \quad \forall\, p \in products$$

```
# Create an object of type list to store the constraints for each product

pro = f.addConstrs(((sum(bom[r,p]*shadowPrice[r] for r in resources) >= price[p]) for p in products), name='V')
```

```
f.setObjective(shadowPrice.prod(capacity))
```

GRB.MINIMZE is the default

Coefficients in the objective function are the resource capacities

Furniture Dual Problem: Solved with Gurobi ..3

```python
# save model for inspection
f.write('furnitureDual.lp')
```

```
\ Model Furniture
\ LP format - for model browsing. Use MPS format to capture full model detail.
Minimize
  400 price[mahogany] + 450 price[labor]
Subject To
 V[table]: 20 price[mahogany] + 15 price[labor] >= 80
 V[chair]: 5 price[mahogany] + 10 price[labor] >= 45
Bounds
End
```

Furniture Dual Problem: Solved with Gurobi ..4

```
# run optimization engine
f.optimize()
```

```
Optimize a model with 2 rows, 2 columns and 4 nonzeros
Coefficient statistics:
  Matrix range     [5e+00, 2e+01]
  Objective range  [4e+02, 5e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [5e+01, 8e+01]
Presolve time: 0.13s
Presolved: 2 rows, 2 columns, 4 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    0.0000000e+00   1.562500e+01   0.000000e+00      0s
       2    2.2000000e+03   0.000000e+00   0.000000e+00      0s

Solved in 2 iterations and 0.16 seconds
Optimal objective  2.200000000e+03
```

Gurobi solver finds the optimal solution of the Furniture dual problem

```
# display optimal values of decision variables
for v in f.getVars():
    if (abs(v.x) > 1e-6):
        print(v.varName, v.x)

# display optimal total revenue value
print('total revenue', f.objVal)
```

```
('price[mahogany]', 0.9999999999999996)
('price[labor]', 4.000000000000001)
('total revenue', 2200.0000000000005)
```

$1.00 The optimal value of the shadow price for mahogany is $1.00

$4.00 The optimal value of the shadow price for labor is $4.00

$2,200 The optimal objective function value is $2,200

Furniture Dual Problem: Solved with Gurobi .. 5

```
# display shadow prices of product constraints
for p in pro:
    if (abs(pro[p].Pi) > 1e-6):
        print(pro[p].ConstrName, pro[p].Pi)
```

```
('V[table]', 14.0)
('V[chair]', 24.000000000000004)
```

- The “shadow prices” of the products’ constraints are 14 tables and 24 chairs. These are the optimal (make) values of the Furniture primal problem. Notice that in both problems, primal and dual, the optimal objective function value is $2,200. This is not a coincidence!!!

Duality in Linear Programming

Remarks:

- In general, it can be shown that the dual of a dual problem is the primal problem, and that when either problem has an optimal solution, the other problem also has an optimal solution, and the optimal objective function value of both problems is the same.

- Another important feature of duality in linear programming is that the optimal solution of the dual problem is contained in the information provided by the simplex method while solving and finding an optimal solution to the primal problem.

- Duality in linear programming provides a good characterization of optimality conditions that can be exploited computationally to solve LP problems efficiently.

Relationship between primal and dual problems

	Primal (maximize)	Dual (minimize)
1)	i'th constraint ≤	i'th variable ≥ 0
2)	i'th constraint ≥	i'th variable ≤ 0
3)	i'th constraint =	i'th variable unrestricted
4)	j'th variable ≥ 0	j'th constraint ≥
5)	j'th variable ≤ 0	j'th constraint ≤
6)	j'th variable unrestricted	j'th constraint =

Primal

$(1.0)\quad Max\ revenue = 45x_1 + 80x_2$

$(2.0)\quad 5x_1 + 20x_2 \leq 400$ Mahogany

$(3.0)\quad 10x_1 + 15x_2 \leq 450$ Labor

$x_1, x_2 \geq 0$

Chairs Tables

Dual

$(4.0)\quad Min\ Cost = 400w_1 + 450w_2$

$(5.0)\quad 5w_1 + 10w_2 \geq 45$ Chairs

$(6.0)\quad 20w_1 + 15w_2 \geq 80$ Tables

$w_1, w_2 \geq 0$

Mahogany Labor

Remark:
Notice that the relationship between the Furniture primal and dual problems is captured by rows 1) and 4)

Optimality conditions

Linear Programming

Optimality conditions in linear programming .. 1

Consider the Furniture primal and dual problems:

Primal

$$(1.0)\quad Max\ revenue = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400$$ Mahogany

$$(3.0)\quad 10x_1 + 15x_2 \leq 450$$ Labor

$$x_1, x_2 \geq 0$$

Chairs Tables

Dual

$$(4.0)\quad Min\ Cost = 400w_1 + 450w_2$$

$$(5.0)\quad 5w_1 + 10w_2 \geq 45$$ Chairs

$$(6.0)\quad 20w_1 + 15w_2 \geq 80$$ Tables

$$w_1, w_2 \geq 0$$

Mahogany Labor

Furniture primal and dual problems in standard form

$$(1.0)\quad Max\ revenue = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 + h_1 = 400$$ Mahogany

$$(3.0)\quad 10x_1 + 15x_2 + h_2 = 450$$ Labor

$$x_1, x_{2,}\ h_{1,} h_2 \geq 0$$

Chairs Tables

$$(4.0)\quad Min\ Cost = 400w_1 + 450w_2$$

$$(5.0)\quad 5w_1 + 10w_2 - s_1 = 45$$ Chairs

$$(6.0)\quad 20w_1 + 15w_2 - s_2 = 80$$ Tables

$$w_1, w_2 s_1, s_2 \geq 0$$

Mahogany Labor

h_1 is the slack variable of the mahogany constraint

h_2 is the slack variable of the labor constraint

s_1 is the surplus variable of the chairs constraint

s_2 is the surplus variable of the tables constraint

Optimality conditions in linear programming .. 2

Consider the optimal solution of both problems, primal and dual

Primal optimal solution:

x1 = 24, x2 = 14, h1 = 0, h2 = 0

$$(1.0)\ Max\ revenue = 45(x1 = 24) + 80(x2 = 14) = 2200$$

Mahogany

$$(2.0)\ 5(x1 = 24) + 20(x2 = 14) + (h1 = 0) = 400$$ binding

Labor

$$(3.0)\ 10(x1 = 24) + 15(x2 = 14) + (h2 = 0) = 450$$ binding

$$x_1, x_2, h_1, h_2 \geq 0$$

Chairs Tables

Dual optimal solution:

w1 = 1, w2 = 4, s1 = 0, s2 = 0

$$(4.0)\ Min\ Cost = 400(w1 = 1) + 450(w2 = 4) = 2200$$

Chairs

$$(5.0)\ 5(w1 = 1) + 10(w2 = 4) - (s1 = 0) = 45$$

Tables

$$(6.0)\ 20(w1 = 1) + 15(w2 = 4) - (s2 = 0) = 80$$

$$w_1, w_2 s_1, s_2 \geq 0$$

Mahogany Labor

Remarks

- Note that the mahogany and labor constraints are binding, i.e. the slack variables (h1 = h2 = 0).
- Note that the shadow price of the mahogany and labor constraints are positive, i.e. (w1 = 1, w2 = 4).
- Note that the chairs and tables constraints are binding, i.e. the surplus variables (s1 = s2 = 0).
- Note that the “shadow price” of the chairs and tables constraints are positive, i.e. (x1 = 24, x2 = 14).

Optimality conditions in linear programming .. 3

In summary

Primal optimal solution	Dual optimal solution
Chair variable x1 = 24	Chair constraint (is binding) surplus variable s1 = 0
Table variable x2 = 14	Table constraint (is binding) surplus variable s2 = 0
Mahogany constraint (is binding) slack variable h1 = 0	Mahogany shadow price w1 = 1
Labor constraint (is binding) slack variable h2 = 0	Labor shadow price w2 = 4

Complementary (a.k.a. orthogonality) conditions in linear programming optimal solutions

- x*s = 0. At optimality, the product of the decision variables in the primal problem and the associate surplus (slack) variables in the dual problem is always zero.
- h*w = 0. At optimality, the product of the slack (surplus) variables in the primal problem and the associate shadow prices in the dual problem is always zero.

GUROBI OPTIMIZATION

Summary of optimality conditions for linear programming

- A solution (x1=24, x2=14) to the primal problem and a solution (w1=1, w2=4) of the dual problem are optimal, if and only if:

- Primal feasibility: we have found a solution of the primal problem that satisfy all its constraints.
  - The production plan satisfies the mahogany and labor constraints.

$$(2.0)\ 5(x1 = 24) + 20(x2 = 14) = 400$$ Mahogany capacity

$$(3.0)\ 10(x1 = 24) + 15(x2 = 14) = 450$$ Labor capacity

- Dual feasibility: we have found a solution of the dual problem that satisfy all its constraints.
  - The shadow prices associated to the mahogany and labor resources satisfy the price constraints for the chairs and the tables.

$$(5.0)\ 5(w1 = 1) + 10(w2 = 4) = 45$$ Chair price

$$(6.0)\ 20(w1 = 1) + 15(w2 = 4) = 80$$ Table price

GUROBI OPTIMIZATION

Summary of optimality conditions for linear programming

- Complementary (orthogonality) conditions:

- The product of the decision variables in the primal problem and the associate surplus variables in the dual problem is always zero. (**Cost efficient**).

  - Since we are building 24 chairs, the surplus variable of the constraint of the price of chairs is 0. That is, x1*s1=0. This means that the opportunity costs of building chairs is equal to the price of the chair:

$$(5.0)\ 5(w1 = 1) + 10(w2 = 4) - (s1 = 0) = 45$$

  - Since we are building 14 tables, the surplus variable of the constraint of the price of tables is 0. That is, x2*s2=0. This means that the opportunity costs of building tables is equal to the price of the table:

$$(6.0)\ 20(w1 = 1) + 15(w2 = 4) - (s2 = 0) = 80$$

Summary of optimality conditions for linear programming

- The product of the decision variables in the dual problem and the associate slack variables in the primal problem is always zero. **(Resource efficient).**

  - Since the shadow price (opportunity cost) of mahogany is \$1, the slack variable of the mahogany constraint is 0. This means we are using the mahogany resource efficiently and there is no waste.

$$(2.0)\quad 5(x1 = 24) + 20(x2 = 14) + (h1 = 0) = 400$$

  - Since the shadow price (opportunity cost) of labor is \$4, the slack variable of the labor constraint is 0. This means we are using the labor resource efficiently and there is no waste.

$$(3.0)\quad 10(x1 = 24) + 15(x2 = 14) + (h2 = 0) = 450$$

GUROBI OPTIMIZATION

Summary of optimality conditions for linear programming

- The optimal objective function value of the primal problem = the optimal objective function of the dual problem. This mathematical theorem is known as **strong duality**.

$$(1.0)\ Max\ revenue = 45(x1 = 24) + 80(x2 = 14) = 2200$$

$$(4.0)\ Min\ Cost = 400(w1 = 1) + 450(w2 = 4) = 2200$$

Dual Simplex method

Variation of Simplex method

The Dual simplex method

The key idea of the dual simplex method is to apply the simplex method to the dual problem, but using the canonical form of the primal problem.

Original LP Problem Formulation

$$(1.0)\quad Max\ revenue(z) = 45x_1 + 80x_2$$

$$(2.0)\quad 5x_1 + 20x_2 \leq 400 \quad \text{Mahogany}$$

$$(3.0)\quad 10x_1 + 15x_2 \leq 450 \quad \text{Labor}$$

$$x_1, x_2 \geq 0$$

Tables

LP Problem Formulation in Canonical Form

$$(1.0)\quad Max\ z = 2400 - {}^{25}/_{3}\, x1 - {}^{80}/_{15}\, h2$$

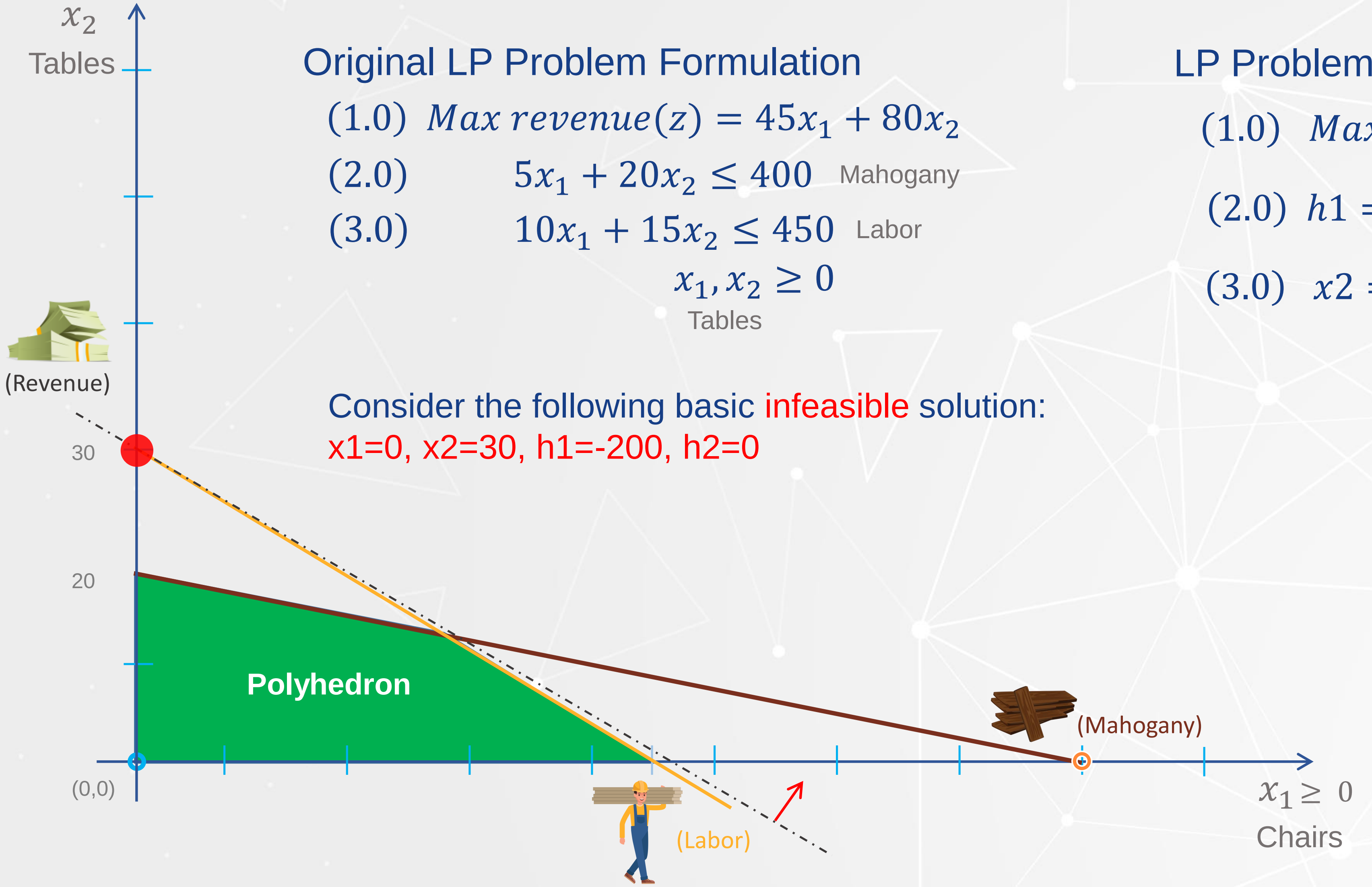
$$(2.0)\quad h1 = -200 + {}^{25}/_{3}\, x1 + {}^{4}/_{3}\, h2$$

$$(3.0)\quad x2 = 30 - {}^{2}/_{3}\, x1 - {}^{1}/_{15}\, h2$$

$$x1, x2, h1, h2 \geq 0$$

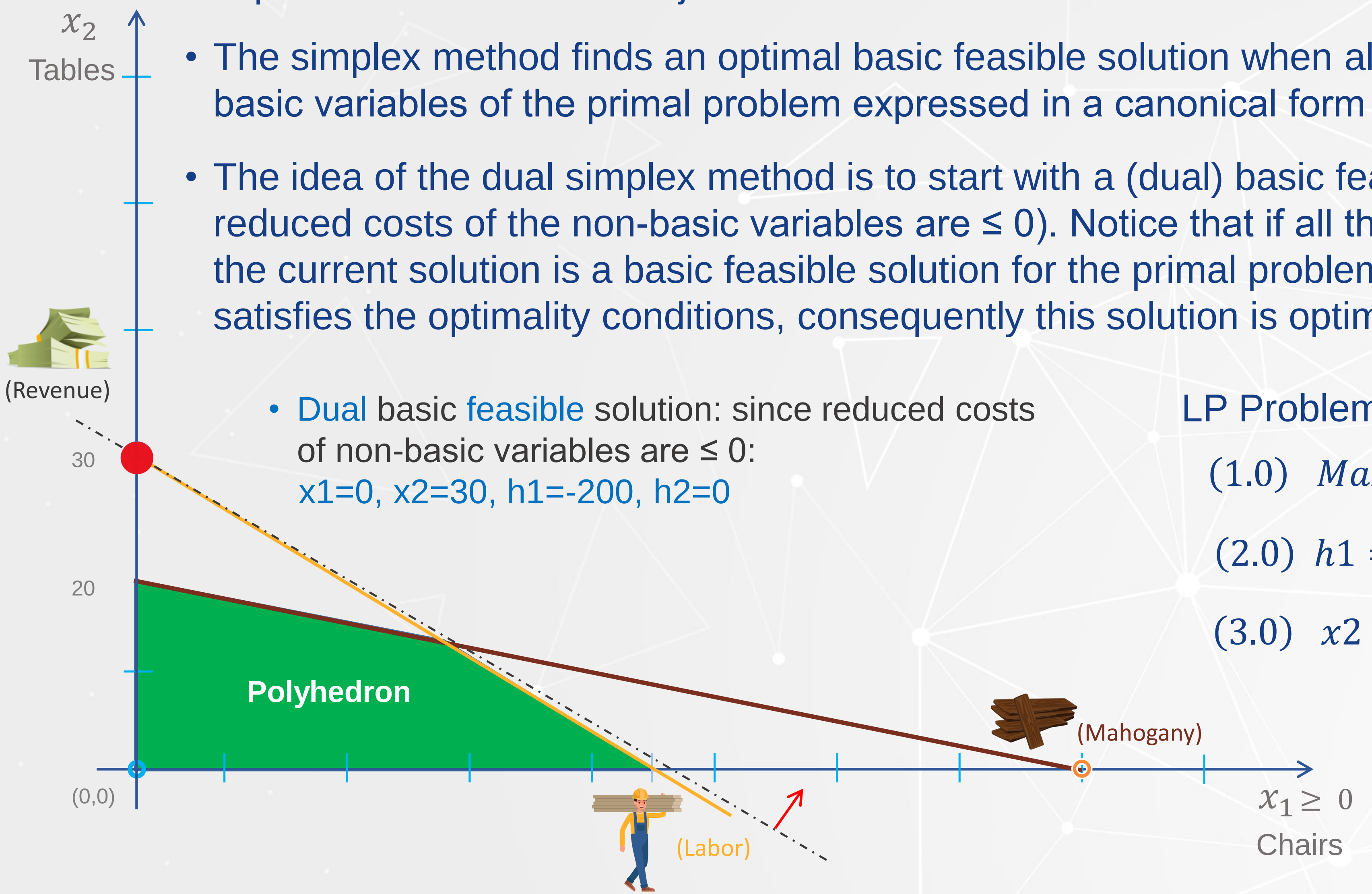
Consider the following basic infeasible solution:

x1=0, x2=30, h1=-200, h2=0



The Dual simplex method ..2

- Recall that the simplex method iterates from one basic feasible solution to another, always trying to improve the value of the objective function.
- The simplex method finds an optimal basic feasible solution when all the reduced costs of the non-basic variables of the primal problem expressed in a canonical form are ≤ 0. (Maximization problem).
- The idea of the dual simplex method is to start with a (dual) basic feasible solution (i.e. all the reduced costs of the non-basic variables are ≤ 0). Notice that if all the basic variables are ≥ 0, then the current solution is a basic feasible solution for the primal problem in a canonical form that satisfies the optimality conditions, consequently this solution is optimal.



- Dual basic feasible solution: since reduced costs of non-basic variables are ≤ 0:
  x1=0, x2=30, h1=-200, h2=0

LP Problem Formulation in Canonical Form

$$(1.0)\quad Max\ z = 2400 - {}^{25}/_{3}\, x1 - {}^{80}/_{15}\, h2$$

$$(2.0)\quad h1 = -200 + {}^{25}/_{3}\, x1 + {}^{4}/_{3}\, h2$$

$$(3.0)\quad x2 = 30 - {}^{2}/_{3}\, x1 - {}^{1}/_{15}\, h2$$

$$x1, x2, h1, h2 \geq 0$$

The Dual simplex method ..3

- Suppose that at least there is one basic variable that is negative. Choose the basic variable with the most negative value as the variable to leave the basis. In this example, the variable h1 will leave the basis.
  - If all the coefficients of the non-basic variables in the canonical form equation of the basic variable leaving the basis are negative, STOP the LP problem is infeasible.

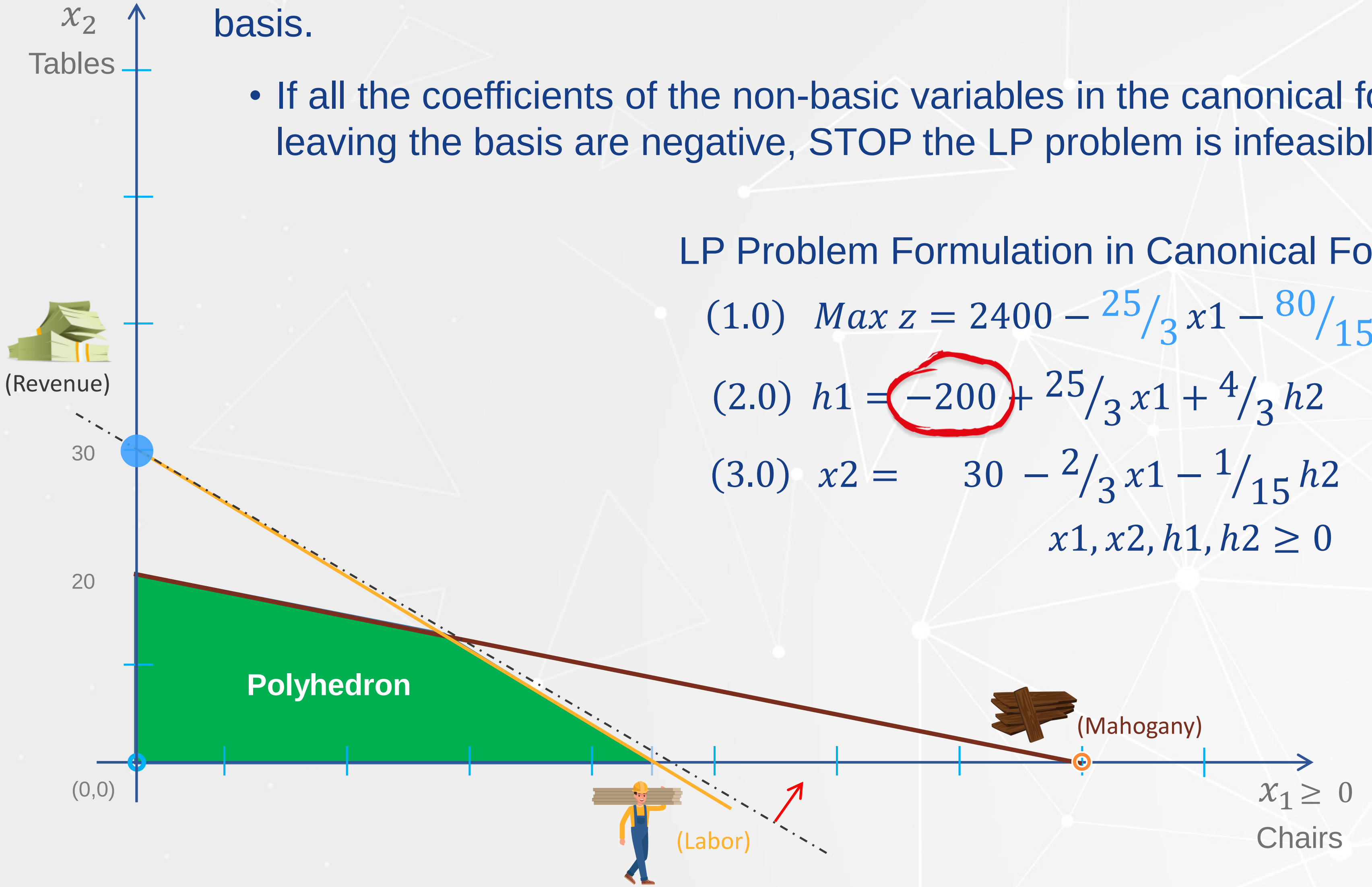
LP Problem Formulation in Canonical Form

$$(1.0)\quad Max\ z = 2400 - {}^{25}/_{3}\, x1 - {}^{80}/_{15}\, h2$$

$$(2.0)\quad h1 = -200 + {}^{25}/_{3}\, x1 + {}^{4}/_{3}\, h2$$

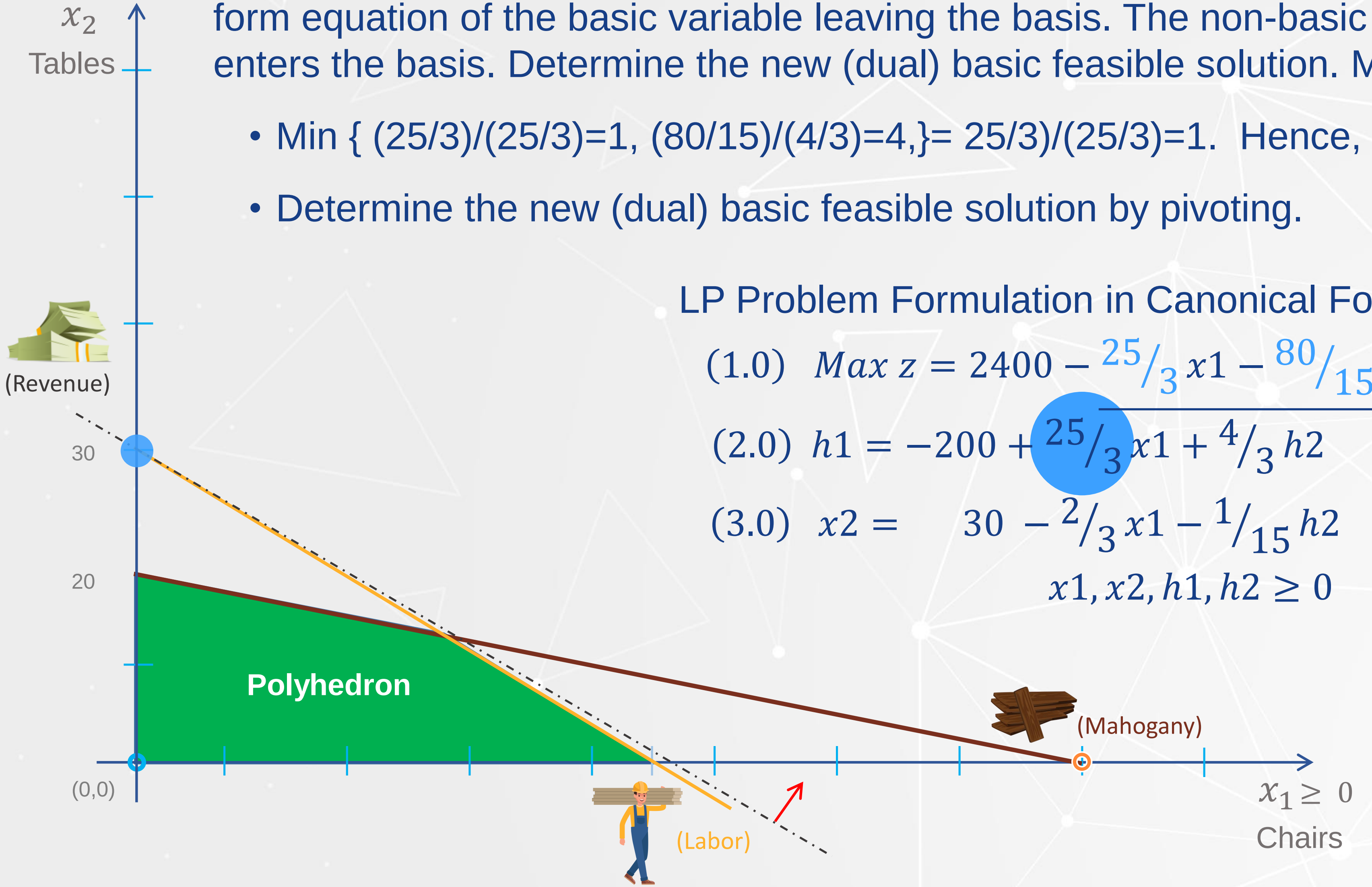
$$(3.0)\quad x2 = \quad 30 - {}^{2}/_{3}\, x1 - {}^{1}/_{15}\, h2$$

$$x1, x2, h1, h2 \geq 0$$



The Dual simplex method ..4

- Else, there is at least one non-basic variable with a positive coefficient, consider the minimum ratio test of the absolute of the reduce costs and the positive coefficient of non-basic variables in the canonical form equation of the basic variable leaving the basis. The non-basic variable with the minimum ratio enters the basis. Determine the new (dual) basic feasible solution. Minimum ratio test:
  - Min { (25/3)/(25/3)=1, (80/15)/(4/3)=4,}= 25/3)/(25/3)=1. Hence, the variable x1 enters the basis.
  - Determine the new (dual) basic feasible solution by pivoting.



LP Problem Formulation in Canonical Form

$$(1.0)\quad Max\ z = 2400 - {}^{25}/_{3}\,x1 - {}^{80}/_{15}\,h2$$

$$(2.0)\quad h1 = -200 + {}^{25}/_{3}\,x1 + {}^{4}/_{3}\,h2 \quad \longrightarrow \text{Pivot}$$

$$(3.0)\quad x2 = 30 - {}^{2}/_{3}\,x1 - {}^{1}/_{15}\,h2$$

$$x1, x2, h1, h2 \geq 0$$

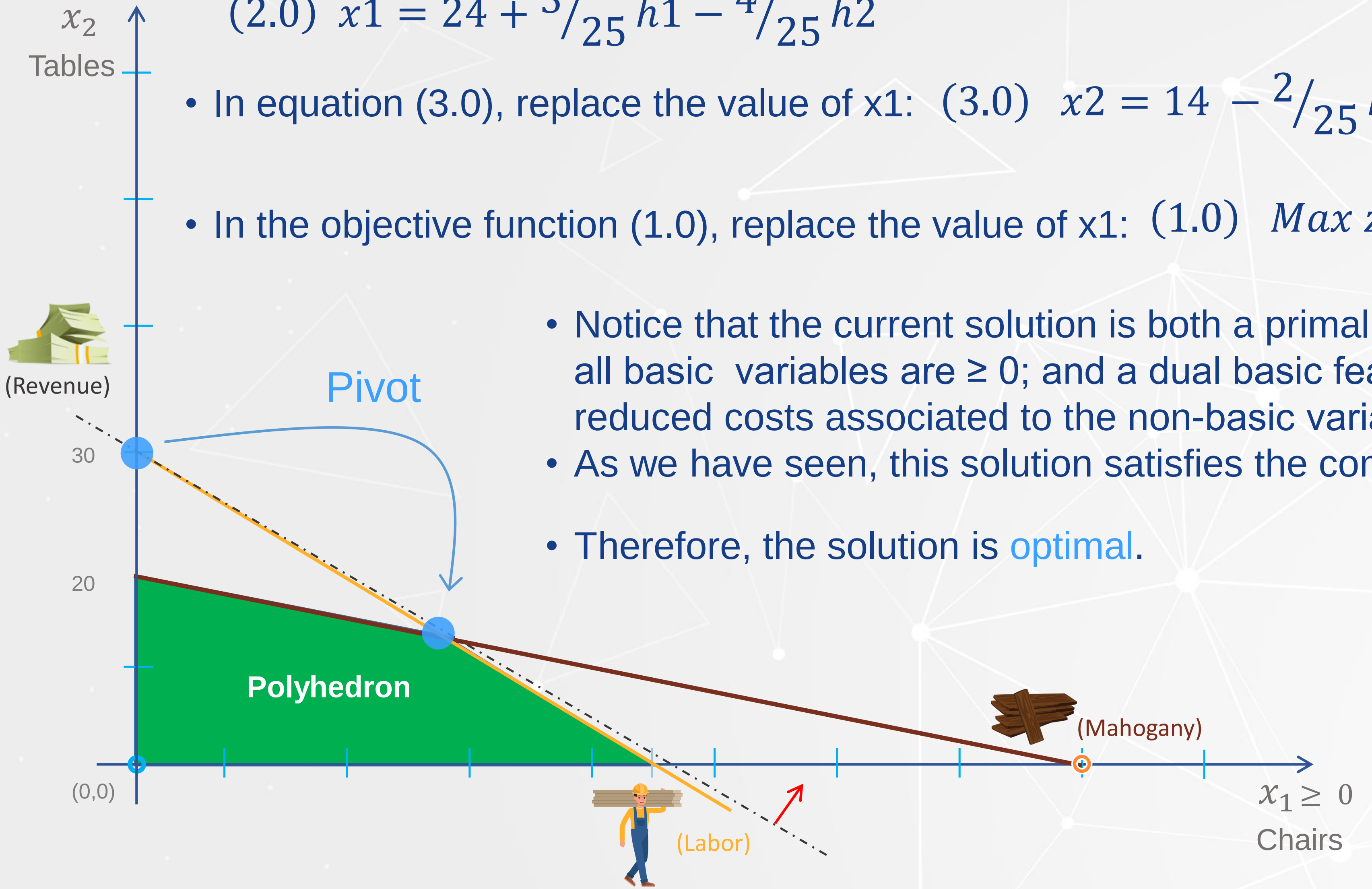
The Dual simplex method ..5

- Pivoting). In equation (2.0), express basic variable x1 in terms of non-basic variables h1 and h2:

$$(2.0)\quad x1 = 24 + {}^{3}/_{25}\, h1 - {}^{4}/_{25}\, h2$$

- In equation (3.0), replace the value of x1: $(3.0)\quad x2 = 14 - {}^{2}/_{25}\, h1 - {}^{1}/_{25}\, h2 \qquad x1, x2, h1, h2 \geq 0$

- In the objective function (1.0), replace the value of x1: $(1.0)\quad Max\ z = 2200 - 1h1 - 4h2$



- Notice that the current solution is both a primal basic feasible solution, since all basic variables are ≥ 0; and a dual basic feasible solution, since all the reduced costs associated to the non-basic variables are ≤ 0.
- As we have seen, this solution satisfies the complementary conditions:

- Therefore, the solution is optimal.

GUROBI OPTIMIZATION

The Dual simplex method ..5

- The dual simplex method is recommended for LP problems in which a dual basic feasible solution is available.

- The dual simplex method is particularly useful for reoptimizing an LP problem after a constraint has been added, since we don’t need to start the solution approach from scratch.

- The cutting planes method to solve MIP problems relies heavily on the dual simplex. The core idea of the cutting planes method is to add a constraint to the LP problem continuous relaxation whenever an integer variable has a fractional value in the optimal solution. The added constraint will make this fractional value of the integer variable infeasible.

END