Applied Computer Science: MAT 3100

OPERATIONS RESEARCH

Mrs. Nafy Aidara
Foreword

The African Virtual University (AVU) is proud to participate in increasing access to education in African countries through the production of quality learning materials. We are also proud to contribute to global knowledge as our Open Educational Resources are mostly accessed from outside the African continent.

This module was developed as part of a diploma and degree program in Applied Computer Science, in collaboration with 18 African partner institutions from 16 countries. A total of 156 modules were developed or translated to ensure availability in English, French and Portuguese. These modules have also been made available as open education resources (OER) on oer.avu.org.

On behalf of the African Virtual University and our patron, the African Development Bank, I invite you to use this module in your institution, for your own education, to share it as widely as possible and to participate actively in the AVU communities of practice of your interest. We are committed to be on the frontline of developing and sharing Open Educational Resources.

The African Virtual University (AVU) is a Pan African Intergovernmental Organization established by charter with the mandate of significantly increasing access to quality higher education and training through the innovative use of information communication technologies. A Charter, establishing the AVU as an Intergovernmental Organization, has been signed so far by nineteen (19) African Governments - Kenya, Senegal, Mauritania, Mali, Cote d’Ivoire, Tanzania, Mozambique, Democratic Republic of Congo, Benin, Ghana, Republic of Guinea, Burkina Faso, Niger, South Sudan, Sudan, The Gambia, Guinea-Bissau, Ethiopia and Cape Verde.

The following institutions participated in the Applied Computer Science Program: (1) Université d’Abomey Calavi in Benin; (2) Université de Ouagadougou in Burkina Faso; (3) Université Lumière de Bujumbura in Burundi; (4) Université de Douala in Cameroon; (5) Université de Nouakchott in Mauritania; (6) Université Gaston Berger in Senegal; (7) Université des Sciences, des Techniques et Technologies de Bamako in Mali (8) Ghana Institute of Management and Public Administration; (9) Kwame Nkrumah University of Science and Technology in Ghana; (10) Kenyatta University in Kenya; (11) Egerton University in Kenya; (12) Addis Ababa University in Ethiopia (13) University of Rwanda; (14) University of Dar es Salaam in Tanzania; (15) Université Abdou Moumouni de Niamey in Niger; (16) Université Cheikh Anta Diop in Senegal; (17) Universidade Pedagógica in Mozambique; and (18) The University of the Gambia in The Gambia.

Bakary Diallo
The Rector
African Virtual University
Production Credits

Author
Nafy Aidara

Peer Reviewer
Robert Oboko

AVU - Academic Coordination
Dr. Marilena Cabral

Overall Coordinator Applied Computer Science Program
Prof Tim Mwololo Waema

Module Coordinator
Florence Tushabe

Instructional Designers
Elizabeth Mbasu
Benta Ochola
Diana Tuel

Media Team
Sidney McGregor          Michal Abigael Koyier
Barry Savala             Mercy Tabi Ojwang
Edwin Kiprono            Josiah Mutsogu
Kelvin Muriithi          Kefa Murimi
Victor Oluoch Otieno     Gerisson Mulongo
Copyright Notice

This document is published under the conditions of the Creative Commons Attribution http://creativecommons.org/licenses/by/2.5/

Module Template is copyright African Virtual University licensed under a Creative Commons Attribution-ShareAlike 4.0 International License. CC-BY, SA

Supported By

AVU Multinational Project II funded by the African Development Bank.
# Table of Contents

## Foreword
- Page 2

## Production Credits
- Page 3

## Copyright Notice
- Page 4

## Supported By
- Page 4

## Course Overview
- Page 7

  - Welcome to Operations Research ............................................. 7
  - Prerequisites ................................................................. 7
  - Materials ..................................................................... 7
  - Course Goals ............................................................... 7
  - Units ............................................................................. 8
    - Assessment ................................................................... 9
  - Schedule ......................................................................... 9
  - Readings and Other Resources ........................................... 9

## Unit 0. Pre-Assessment
- Page 13

  - Unit Introduction ............................................................ 13
  - Unit Objectives .............................................................. 13
  - Key Terms ..................................................................... 13
  - Unit Assessment ............................................................ 14
    - Instructions .................................................................. 19
    - Grading Scheme ......................................................... 19
    - Feedback ..................................................................... 19
  - Unit Readings and Other Resources ................................ 19

## Unit 1. Linear Programming, the Simplex Method
- Page 20

  - Unit Introduction ............................................................ 20
  - Unit Objectives .............................................................. 20
  - Key Terms ..................................................................... 20
  - Learning Activities ....................................................... 22
    - Activity 1 Linear Programming Problem Formulation ........ 22
<table>
<thead>
<tr>
<th>Activity Details</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>22</td>
</tr>
<tr>
<td>Conclusion</td>
<td>35</td>
</tr>
<tr>
<td>Assignment</td>
<td>35</td>
</tr>
<tr>
<td>Activity 2 The Simplex table method</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>38</td>
</tr>
<tr>
<td>Activity 3 Simplex Table Algorithm</td>
<td>41</td>
</tr>
<tr>
<td>Conclusion</td>
<td>47</td>
</tr>
<tr>
<td>Activity 4 SIMPLEX with spreadsheet and SOLVER</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>47</td>
</tr>
<tr>
<td>Conclusion</td>
<td>48</td>
</tr>
<tr>
<td>Assignment</td>
<td>49</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unit Readings and Other Resources</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unit 2. Duality and Sensitivity Analysis</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td>Unit Introduction</td>
<td>51</td>
</tr>
<tr>
<td>Unit Objectives</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>51</td>
</tr>
<tr>
<td>Key words</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>52</td>
</tr>
<tr>
<td>Learning Activities</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>52</td>
</tr>
<tr>
<td>Activity 2.1 - Existence of solutions and cases in the implementation of Simplex Introduction</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>52</td>
</tr>
<tr>
<td>Activity details</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>57</td>
</tr>
<tr>
<td>Conclusion</td>
<td>57</td>
</tr>
<tr>
<td>Assessment</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>58</td>
</tr>
<tr>
<td>Activity 2.2 - Big M and the Two Phases</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>58</td>
</tr>
<tr>
<td>Activity Details</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>62</td>
</tr>
<tr>
<td>Conclusion</td>
<td>62</td>
</tr>
<tr>
<td>Assessment</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>64</td>
</tr>
<tr>
<td>Activity 2.3 - Duality</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>64</td>
</tr>
<tr>
<td>Activity Details - The dual problem</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>71</td>
</tr>
<tr>
<td>Conclusion</td>
<td>71</td>
</tr>
<tr>
<td>Evaluation</td>
<td>72</td>
</tr>
</tbody>
</table>
# Unit 3. Transport, Assignment Problems, and Network Flow

## Introduction to unit

84

## Unit Objectives

84

## Key Terms

84

## Learning activities

87

- Activity 3.1 - Terminology and basic elements of the graph theory

87

## Activity Details

88

- Conclusion

95

- Activity 3.2 - Transport Problems

95

## Activity Details

96

- Método MODI

98

- Method of the upper left corner

98

- Method of minimum cost

99

- Method of penalties (or In Vogel)

99

- Alternating Series (Series)

100

- The stepping stone method

101

- Hidden Variable

102

- Conclusion

107

- Activity 3.3 - Assignment problem

107

## Activity details

107

- Conclusion

112

- Assignment

112

## Unit Summary

112

## Unit Readings and Other Resources

112
Course Overview

Welcome to Operations Research

Operations research analyzes and presents scientific procedures for decision making in engineering, technology management and management human, productive and financial resources of an organization.

The main purpose of Operations Research is to optimize the system performance when subjected to restrictions.

It is based on mathematical modeling and uses optimization techniques (convex) and graph theory.

The main calculation algorithm is the SIMPLEX table and network, considered one of the two most important algorithms of the twentieth century (https://www.siam.org/pdf/news/637.pdf).

Operations research was meant to optimize the use of scarce resources during the 2nd world war and, having gained dominance in the industry and management. It is presented today as one of the main tools of business decision making (scientific and non-empirically).

In addressing this matter, the main priority is the “basic principles” and detailed examples at the expense of theoretical complexity that this subject today presents.

O.R. can be used in optimal radiation therapy, finding the shortest path in a network, setting prices in the stock exchange market, etc.

Prerequisites


Materials

The materials required to complete this course are:

Access to Excel / Solver or Google / Sheets / Solver -https://www.google.com/sheets/about/

Introduction to Operations Research, Fernando Marins, UNESP, São Paulo, 2011-

Introduction to Operations Research, 8th Ed, Frederick Hillier, Gerard Lieberman, McGraw, 2006;


Introduction to Probability and Statistics, Maria G Martins, 2005 -

Course Goals

Upon completion of this course the learner should be able to:

build quantitative models of a management problem, formulate a linear optimization problem, use the Simplex method to manually solve simple linear programming problems, solve complex linear programming problems using SOLVER model and solve a transportation
problem and designation / assignment; model flows in the network and find the path of least weight, the maximum flow and minimum cost of a stream, model a waiting list problem and determine the average time and the average length of a queue; model simple cases of project planning and the game theory

Units

Unit 0: Pre-Assessment

Makes a review of the main concepts of linear algebra and analytic geometry necessary to following units; uses some statistical concepts to determine the statistical parameters the distribution. Poisson (number of events in a given time interval), the exponential distribution (time between two events) and binomial distribution; Use spreadsheet software to implement automatic calculation routines.

Unit 1: Linear programming, Simplex method

Displays concepts of mathematical modeling and formalization of linear programming problems, presents the terminology and basic elements of linear optimization method, presents the computational foundations of the Simplex method from a two-dimensional problem, illustrating the resolution with graphic and algebraic method; addresses some of maximization-type problems using tabular SIMPLE algorithm in a spreadsheet; introduces the add-on Google Sheets, SOLVER.

Unit 2: Linear Programming, Duality and sensitivity analysis

Presents both Simplex algorithm variants to minimize the target function: “big M” and “two phases”;

Displays the linear programming theorem, treating the question of the existence and uniqueness of solutions;

It startup conditions, iteration and the end of the Simplex algorithm, and solves some problems-type linear programming making use of spreadsheets and SOLVER;

Displays the relationship between the primal and the dual of a linear programming problem, the fundamentals of sensitivity analysis and illustrates the concepts with some examples;

Unit 3 Transport Problems and designation

Presents the basics and terminology of graph theory,

Presents the problem of transport and some algorithms for its resolution

Presents the problem of designation and the Hungarian algorithm for its resolution problems.

Unit 4 Network Optimization and Queue

Represents optimization of networks analyzes the shortest path, network flow and minimum cost flow network
presents the basics and terminology of stochastic processes Markov, features elements of Markov processes and the theory of queues, studies the project planning cases and game theory.

**Assessment**

Formative assessments, used to check learner progress, are included in each unit.

Summative assessments, such as final tests and assignments, are provided at the end of each module and cover knowledge and skills from the entire module.

Summative assessments are administered at the discretion of the institution offering the course. The suggested assessment plan is as follows:

<table>
<thead>
<tr>
<th></th>
<th>Unit 1 Assignment</th>
<th>15%</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Unit 2 Assignment</td>
<td>15%</td>
</tr>
<tr>
<td>3</td>
<td>Unit 3 Assignment</td>
<td>15%</td>
</tr>
<tr>
<td>3</td>
<td>Unit 4 Assignment</td>
<td>15%</td>
</tr>
<tr>
<td>4</td>
<td>Module Final Assessment</td>
<td>40%</td>
</tr>
</tbody>
</table>

**Schedule**

<table>
<thead>
<tr>
<th>Unit</th>
<th>Activities</th>
<th>Estimated time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Preliminaries + Evaluation</td>
<td>20 hours</td>
</tr>
<tr>
<td>1</td>
<td>Theoretical activities+Practical exercises+formative Evaluation</td>
<td>25 hours</td>
</tr>
<tr>
<td>2</td>
<td>Theoretical activities+Practical exercises+formative Evaluation</td>
<td>25 hours</td>
</tr>
<tr>
<td>3</td>
<td>Theoretical activities+Practical exercises+formative Evaluation</td>
<td>25 hours</td>
</tr>
<tr>
<td>4</td>
<td>Theoretical activities+Practical exercises+formative Evaluation</td>
<td>25 hours</td>
</tr>
<tr>
<td>5</td>
<td>Final Assessment</td>
<td>20 hours</td>
</tr>
</tbody>
</table>

**Readings and Other Resources**

The readings and other resources in this course are:
Unit 0

Required readings and other resources:

Application Worksheet - https://www.google.com/sheets/about/
Khan Academy - https://pt.khanacademy.org/

Optional readings and other resources:
https://pt.wikipedia.org/wiki/%C3%81lgebra_linear
https://en.wikipedia.org/wiki/Poisson_distribution
https://en.wikipedia.org/wiki/Exponential_distribution

Unit 1

Required readings and other resources:

Operations Research, Prof. G. Srinivasan, NPTEL;
Application Worksheet - https://www.google.com/sheets/about/
Khan Academy - https://pt.khanacademy.org/

Optional readings and other resources:
Linear and Non-Linear Programming, 3rd Ed, David Luenberger, Yinyu Ye, Springer, 2008;
International Ferderation of Operational Research Societies - http://ifors.org/web/
The Institute for Operations Research and the Management Sciences http://www.informs.org,

Unit 2

Required readings and other resources:

Course Overview

Operations Research, Prof. G. Srinivasan, NPTEL;
Application Worksheet - https://www.google.com/sheets/about/
Khan Academy - https://pt.khanacademy.org/
Optional readings and other resources:
Linear and Non-Linear Programming, 3rd Ed, David Luenberger, Yinyu Ye, Springer, 2008;
International Ferderation of Operational Research Societies - http://ifors.org/web/
The Institute for Operations Research and the Management Sciences http://www.informs.org;

Unit 3
Required readings and other resources:
Operations Research, Prof. G. Srinivasan, NPTEL;
Application Worksheet - https://www.google.com/sheets/about/
Khan Academy - https://pt.khanacademy.org/
Optional readings and other resources:
Linear and Non-Linear Programming, 3rd Ed, David Luenberger, Yinyu Ye, Springer, 2008;
International Ferderation of Operational Research Societies - http://ifors.org/web/
The Institute for Operations Research and the Management Sciences http://www.informs.org;

Unit 4
Required readings and other resources:
Operations Research, Prof. G. Srinivasan, NPTEL;
Application Worksheet - https://www.google.com/sheets/about/
Khan Academy - https://pt.khanacademy.org/

Optional readings and other resources:
Linear and Non-Linear Programming, 3rd Ed, David Luenberger, Yinyu Ye, Springer, 2008;
International Federation of Operational Research Societies - http://ifors.org/web/
The Institute for Operations Research and the Management Sciences http://www.informs.org
Unit 0. Pre-Assessment

Unit Introduction

The purpose of this unit is to check the understanding of linear algebra knowledge, analytic geometry and statistics which owns and that are related to this course.

Unit Objectives

Upon completion of this unit you should be able to:

- apply previous knowledge of algebra and analytical geometry in solving problems;
- apply previous knowledge of statistics to determine the statistical parameters of the Poisson distribution (number of events in a given time interval), exponential (time between two events) and binomial distribution; use spreadsheet software to implement calculation routines.

Key Terms

- **vector space**: scalar body acting on an Abelian group (vectors)
- **Linear combination**: sum of scalar multiplication of vectors
- **Linear Independence**: zero linear combination of vectors is trivial or zero
- **Generator**: extension by linear combination
- **Base**: set of linearly independent vectors and generators
- **vector Subspace**: subset that is also a vector space
- **direct sum**: subspaces whose sum is all the space and whose intersection is the trivial space \{0\}
- **Linear Transformation**: function that preserves linear combination
- **void space**: subspace vectors whose image is the zero vector
- **Matrix**: arrangement of numbers in rows matrix
- **Matrix multiplication**: sum of columns entries in a particular order
- **Domestic product order**: matrix multiplication row with matrix column (SUMPRODUCT)
- **Linear form**: linear function group abelian to the body of a vector space
- **Dual space**: the space of the linear forms in a vector space;
  - Line: level set of a linearly R2
  - Plan: level set in a linear manner in R3
Poisson distribution: \( P(X = n) = \left( e^{-\lambda} \cdot \frac{\lambda^n}{n!} \right) \)

Binomial distribution: \( P(Y = k) = \frac{n}{k} \cdot p^k \cdot (1-w)^{nk} \)

Unit Assessment

Check your understanding!

Question 1

Given the matrices

\[
A = \begin{bmatrix} 3 & 0 & -2 \\ \end{bmatrix} \\
B = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix} \\
C = \begin{bmatrix} 0 & -3 & 1 \\ 1 & -1 & -1 \end{bmatrix} \\
D = \begin{bmatrix} 2 & 1 \\ 1 & 0 \\ -3 & -1 \end{bmatrix} \\
E = \begin{bmatrix} 3 & 4 & 3 \\ 3 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}
\]

a. Calculate

\[
AB = \\
BA = \\
CD = \\
AE = \\
AC^T = \\
EB = \\
DC = \\
B^TB = \\
AD = \\
\]

b) Explain why it is not possible to calculate \( A + B, AC, BC, DE \) or \( EC \)

c) Calculate \( [A]_{12} + [B]_{31} - [C]_{23} + [D]_{12} - [E]_{22} \)

d) Use the Gaussian elimination algorithm to find the staggered matrices \( C, D \) and

e) Using the Gauss-Jordan elimination to find the inverse \( E \) and resolve the equation

\( Ex = B \)
Question 2

a) With respect to the canonical basis of \( \mathbb{R}^2 \), what is the equation of the line through the points P: (4,0) and Q: (0.3)

b) What is the equation segment between P, Q?

c) What are the coordinates of point that is half of P, Q?

d) What are the coordinates of the line intersection point in (a) with the line \( 2y-x = 1 \)

Question 3

a) for the canonical base \( \mathbb{R}^3 \), which the plane equation passing through the points Q: (4,0,0), Q: (0,3,0) and R: (0,0, -1)

b) What point coordinates which is the middle of the triangle vertices P, Q and R

c) What is the equation of the straight line resulting from the intersection of the surface of (a) with the surface \( x-2y + z = 1 \)

Question 4

An insurance company receives on average one accident per hour of participation. Take the Poisson distribution and admit that the company works on average six hours a day, five days a week.

A) How likely is an hour not occur any accident participation?

b) What proportion of days with less than three holdings?

c) How likely is three days a week, there is daily participation of four accidents?

Question 5

In a process (without memory) Markov, the state vector \( P^k \) generating k depends only on the previous generation \( P^{k-1} \) using the relation \( P^k = TP^{k-1} \) where entries \( [T]_{ij} \) on T transition matrix measure the state probability changing i to state j in a generation.

Consider a Markov system with a population divided into three states, with the transition matrix

\[
E = \begin{bmatrix}
1/2 & 1/4 & 0 \\
1/2 & 1/2 & 1/2 \\
0 & 1/4 & 1/2
\end{bmatrix}
\]

and initial state

\[
P^0 = \begin{bmatrix}
1/3 \\
1/3 \\
1/3
\end{bmatrix}
\]
Explain why the entries $T$ transition matrix are not negative, the sum of entries in any of $T$ columns is 1 and the sum of $P_0$ entries also is one?

b) What is the state vector of the 3rd generation, $P_3$?

Solution

Question 1

a) 
\[
A = 4
\]
\[
BA = \begin{bmatrix}
6 & 0 & -4 \\
-3 & 0 & 2 \\
3 & 0 & -2
\end{bmatrix}
\]
\[
CD = \begin{bmatrix}
-6 & -1 \\
4 & 2
\end{bmatrix}
\]
\[
DC = \begin{bmatrix}
1 & -7 & 1 \\
0 & -3 & 1 \\
-1 & 10 & -2
\end{bmatrix}
\]
\[
AC^T = \begin{bmatrix}
-2 & 5
\end{bmatrix}
\]
\[
B^TB = 6
\]
\[
AD = \begin{bmatrix}
12 & 5
\end{bmatrix}
\]
\[
AE = \begin{bmatrix}
3 & 2 & 3
\end{bmatrix}
\]
\[
EB = \begin{bmatrix}
3 \\
-1 \\
1
\end{bmatrix}
\]
\[
ED = \begin{bmatrix}
-1 & 0 \\
-7 & 2 \\
-1 & 0
\end{bmatrix}
\]

b) The dimensions of the arrays are not compatible with the operations.

c) 
\[
[A]_{12} + [B]_{31} + [C]_{20} + [D]_{12} + [E]_{22} = 0 + 1 -(-1)(1) - 2 = 0
\]

d) 
\[
C \rightarrow \begin{bmatrix}
1 & -1 & -1 \\
0 & -3 & 1
\end{bmatrix}
\]
\[
D \rightarrow \begin{bmatrix}
1 & 0 \\
0 & 1 \\
0 & 0
\end{bmatrix}
\]
\[
E \rightarrow \begin{bmatrix}
1 & 0 & 1 \\
0 & -1 & 0 \\
0 & 0 & -2
\end{bmatrix}
\]
e) 
\[ E^{-1} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 1 \\ 0 & 0 & -1 \\ \frac{1}{2} & -\frac{1}{2} & -1 \end{bmatrix} \]

\[ x = \begin{bmatrix} \frac{3}{2} \\ \frac{1}{2} \end{bmatrix} \]

Question 2

a) 
\[ 3x + 4y = 12 \]

b) \((1-t)\)

\((4.0) + t (0,3) \) to \(t \in [0,1]\)

\((x, y) = ((1- t) 4,3t)\) parametric equations

for
c) \(t = (1/2)\)

\((x, y) = (2,7 / 2)\)

d) \((x, y) = (2,3 / 2)\)

Question 3

a) \[ 3x + 4y - 12z = 12 \]

b) 
\[ (x,y,z)_{mid} = (4 / 3,1,-1 / 3) \]

c) 
\[ \lambda(24 / 5,24 / 10,1)+(1-\lambda)(2,2 / 3,0) \]}
Question 4

a) by time (unit time)

\( X \sim \text{Po}(1) \) ie the random variable \( X \) has Poisson distribution with parameter \( \lambda = 1 \)

\[
\mathbb{P}(X = n) = \left( \frac{\lambda^n}{n!} \right) \frac{e^{-\lambda}}{n!} = \frac{e^{-\lambda}}{n!} = \frac{e^{-1}}{n!} = 0.36788 = 37\%
\]

b) By day (time unit) the company works for 6 hours

\( X \sim \text{Po}(6) \)

\[
P(X = 0,1,2) = \left( \frac{e^{-6}6^0}{0!} \right) + \left( \frac{e^{-6}6^1}{1!} \right) + \left( \frac{e^{-6}6^2}{2!} \right)
= 6.2\%
\]

c) day (time unit) operates the company for 6 hours

\( X \sim \text{Po}(6) \)

\[
P(X = 4) = \left( \frac{e^{-6}6^4}{4!} \right) = 54e^{-6} = 0.134
\]

\( Y \) is the number of days in a week, with a total of 5 days in a week

\( Y \sim \text{B}(5; 0.134) \) ie the random variable has a binomial distribution with mean 0.134

\[
P(Y = k) = \binom{n}{k} p^k (1-p)^{n-k}
\]

\[
P(Y = 3) = \binom{5}{3} (0.134)^{3} (1-0.134)^{2} = 0.018 = 1.8\%
\]

a) correspond to transition probabilities of state \( i \) to state \( j \) logo are real values in \([0,1]\)

The sum of the entries in any of the columns is 1 because the sum of the probabilities for transit from any of the states for a given state is 1

Total units is equal to the sum of the initial units of each of the states.

b) \( T^3 = TP^2 = T(TP^1) = T^2 (TP^0) = T^3P^0 \)

\[
T^3 = \begin{bmatrix}
\frac{1}{4} & \frac{1}{2} & \frac{1}{4}
\end{bmatrix}
\]
Instructions
The test has five questions to be answered in a time of 120 minutes.

Grading Scheme
Each question has a weight of 20 points for a total of 100 points.

Feedback
With results above 50%, students can continue with the remaining units.

With a result below 50%, the student must do a review of the modules of Linear Algebra, Analytical Geometry and Statistics before continuing.

Unit Readings and Other Resources
The readings in this unit are to be found at the course-level section “Readings and Other Resources.”
Unit 1. Linear Programming, the Simplex Method

Unit Introduction

Operations research provides tools on to how to work and produce efficiently, to move items (logistics), to reduce queue length during working hours and also do project management efficiently. Students will learn how to solve real problems occurring in real life since most major advances in Operations Research have come from work on real problems.

This unit introduces the concepts of mathematical modeling and formalization of linear programming problems. This unit deals with the terminologies and the basic elements of linear optimization approach. The foundation of the computational of the Simplex method from a two-dimensional problem are presented through the graphic and algebraic method.

Solutions of the Minimum and Maximization problems using the Simplex algorithm is introduced; so is the use of SOLVER to solve such problems.

Unit Objectives

Upon completion of this unit, you should be able to:

- Develop simple mathematical models relating to real or idealized processes;
- Formulate linear programming problems and present them in standard form;
- Describe the linear optimization problem chart of a method in two dimensions;
- Apply previous knowledge of algebra and descriptive geometry in a simple linear programming problem solving of two dimensions;
- Manually apply the Simplex algorithm to solve simple linear programming problems;
- Use SOLVER to solve simple linear programming problems;

Key Terms

- **Schedule**: planning
- **Modeling**: problems of representation in different areas (numbers)
- **Formulation**: presentation of the numerical model in a certain way
- **System**: the focus of the analysis (the observer)
- **Input**: striking a system
- **Output**: system response to (s) impact (s)
- **Process**: evolution of the system in time
Value: resulting monetary amount of the balance between supply and demand

Yield: ratio between input units per unit of output

Decision variables: attributes that define system performance

Objective function: the attribute (dependent) to be optimized

Constraints: constraints on system performance

Optimization: maximum performance within the system operation area

Convex: the straight points between two points are in the set

Linear: proportional relationship

Feasible region: System operating region

Basic solution: vertices of the intersection region of some restrictions

Basic feasible solution: vertices of the intersection region of the restrictions

Great solution: vertex where the value of the objective function is maximum

Slack variables: variables that compensate for inequalities

Basic variable: variables that are calculated on the basis of other

Non-basic variable: free variables that are reset to zero

Simplex: optimization problem solving procedure

Initialization: the initial values entered in the algorithm

Iteration: Run-control commands of the algorithm

End: end conditions of the algorithm

SOLVER: optimization problem solving software
Learning Activities

Activity 1 Linear Programming Problem Formulation

Introduction

It presents the general concepts of mathematical modeling and the most common decision models:

- Programming (in the sense of planning)
- Network flow
- Queues
- Control

It presents the stages of modeling and the formulation of linear programming problems, with more frequent examples [Marins]:

- Mix production
- Advertising campaign
- Staff training
- Mix in Chemical industry
- Machines use Load Balancing
- Scaling teams

Activity Details

Models decision

The objective of operations research is the “modeling and decision making in real systems, deterministic or probabilistic, concerning the need to allocate scarce resources.” [Marins]

Models are projections (representations) of a reality / concept in a particular “space”.

For example, a paper plane is a projection of a concept of a paper material space.

A digital plan of a house is the projection of a concept in a geometric space of points, lines, areas and volumes.

Mathematical modeling arises naturally in the human activity.

Example:

As an example, the simple activity to share a bag of rice between neighbors is modeled naturally by quantifying the rice in the bag (12 kg) and the amount of neighbors (6). This measurement allows to solve a problem (12/6 = 2) optimizing the peace among neighbors: each neighbor gets 2 kg of rice.
There are always two sides of the same question: who gives the rice (primal) and the receiver of the rice (dual).

The modeling aims at making the best decision or the optimal decision.

Of course there are other models of distribution of rice that could take into account the family composition and age of each of the family members or other factors considered relevant.

Therefore modeling is not unique and depends on the level of analytical detail to be achieved.

Today, mathematical modeling is complex realities and concepts in the space of abstract relations expressed by mathematical language.

the Bernoulli distribution $P(X = 1) = p$ serves as a stochastic model of the final state of the coin thrown into the air, where the random variable $X = 1$ if exit “face” and the $p$ parameter determines the probability “a priori” of out “face” (or the plan that the “gods” have already pre-destined to launch).

A system is the observer’s analysis of focus.

The system class ranges from physical systems (particles, mechanical bodies, ...), biological systems (organism, ecosystem, ...), technological systems (devices, cameras, cars, ...), information and communication systems (computers, mobile phones, communication towers, routers, ...), management systems (companies, factories, retail outlets, ...), etc.

Mathematical models are generically applied to systems (organization) and processes (organic evolution).

A system inputs are transformed in a given output, according to the system’s specificity.

In both case the input and the output are time series and can be feed-back lines (or feed-forward), where some or all of the input is returned back as input to better “control output.”

In modeling physical systems, each input and each output is measurable and units of measure are standard in measurement units system is used here International (SI) units of measurement.

<table>
<thead>
<tr>
<th>Measurements</th>
<th>Units</th>
<th>Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>distance</td>
<td>meter</td>
<td>m</td>
</tr>
<tr>
<td>Mass</td>
<td>Kilogram</td>
<td>kg</td>
</tr>
<tr>
<td>Time</td>
<td>Second</td>
<td>s</td>
</tr>
<tr>
<td>Current</td>
<td>Amper</td>
<td>A</td>
</tr>
<tr>
<td>Temperature</td>
<td>Kelvin</td>
<td>K</td>
</tr>
<tr>
<td>Material quantity</td>
<td>Molar</td>
<td>mol</td>
</tr>
<tr>
<td>Light Intensity</td>
<td>Candela</td>
<td>cd</td>
</tr>
</tbody>
</table>
In shaping human resource management systems, productive and financial of a company, the physical units are converted into monetary units through the notion of value.

The current market value allows you to translate different units of measurement in monetary units.

For example, if the input of a given system is 1 kilogram of oil paint is then convertible currencies in 1000 if this is the market value of each liter of oil paint.

As specificity, each system has technical performance parameters that determine the input processing capacity in output.

For example, if a seamstress has a yield of 2 shirts per day, and if there is no lack of raw material, a system with three seamstresses produces 6 shirts.

Yields are typically associated with a certain period of time and are normally categorized into specific technical tables for each system (Example: http://orcamentos.eu/fichas-rendimento/):

<table>
<thead>
<tr>
<th>Item</th>
<th>Descrição</th>
<th>Un</th>
<th>Qté</th>
<th>Preço venda</th>
<th>Valor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Much of the human resource management, production and financial company focus on making the best decision (great decision) in the following decision models

Scheduling problem (planning) - allocation of limited resources to produce outputs competitors

2. Problem of a system flow network - flow through a network of interconnected systems with different "weights" connection, which inputs a system other serves as output:
3. Queuing problem - random input but with accumulation / delay between the input and output

4. Control problem - the output is controlled from the regulation of certain inputs;

   The first three business decision models will be treated throughout this course.

   The control problem is not covered in this course.

Scheduling problem (planning)

   In this class of problems cases involving the following will be analyzed

   Determination of the production mix
   Advertising campaign
   Staff training
   Mix in Chemical industry
   Machines use Load Balancing
   Design of inspection teams and other problems with similar characteristics.

   In most cases, the design and implementation of a programming model (planning) requires the following steps:

   Define the problem;
   Collect data;
   Formulating the mathematical model in terms of an optimization problem;
   Test and refine the model;
   Solve the optimization problem and to simulate different operating conditions;
   Determine the optimal region of operation of the system;
   Implement the operation of the system in the optimal region;
   Controlling the deviation from the optimum system region;

Defining the problem

   The problem definition is necessary to establish very clearly the purpose to be optimized - the objective function whose value is a real number that measures the performance or value of a decision on decision variables.

Data collection

   Data collection allows to identify the decision variables that directly influence the objective function. It helps to determine the restrictions on which the decision variables are subjected to, and get the income and technical parameters that influence the objective function.

Formulation
In the formulation of the model of the optimization, the problem is posed in the form of
\[ f_0(x) \] the objective function subject to \( m \) constraints \[ f_i(x) < b \quad \text{for} \quad i = 1, \ldots, m, \]
where \( x \) is the vector of the decision variables \( x = (x_1, \ldots, x_n) \).

Test

It is important to test the model and tune it in order to objectively represent the actual conditions of the system.

Simulation

The model simulation identifies the variables with the greatest impact on the objective function and treat these variables with particular attention.

Solution

The model solution are to find the vector of decision variable that satisfies all constraints and optimize (maximize or minimize) the objective function.

Control

The system then should be taken to operate at optimal region found and control systems should be introduced in order to draw attention to deviations from the optimal region of operation.

Of linear programming problems formulation technique

In general, an optimization problem is classified into convex or non-convex objective function according to \( f_0(x) \) and \( m \) restrictions \( f_i(x) < b \quad \text{for} \quad i = 1, \ldots, m, \)
where \( x \) is the vector of decision variables \( x = (x_1, \ldots, x_n) \) are convex or not.

Remember that a function is convex if
\[ f(ax + \beta y) \leq af(x) + \beta f(y) \quad \text{para} \quad a + \beta = 1 \quad a, \beta \in [0,1] \]

An important class of convex optimization problems are called linear programming problems, where \( f_0, f_1, \ldots, f_m \) are linear vector functions of the decision variables \( x = (x_1, \ldots, x_n) \).

Remember that a function is linear if:
\[ f(ax + \beta y) = af(x) + \beta f(y) \quad \text{where} \quad a + \beta \in \mathbb{R} \]

The nonlinear functions can be represented by linear combinations of components of the variable vector:
\[ f(x) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \quad \text{where} \quad c_i \in \mathbb{R} \]

The formulation of a linear programming model consists in presenting the problem as an optimization objective function:
Maximization or minimization:

\[ z = c_1x_1 + c_2x_2 + \cdots + c_nx_n \]

subject to constraints:

\[ a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \leq b_1 \]

\[ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \leq b_2 \]

\[ \vdots \]

\[ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \leq b_m \]

and a non-negativity:

\[ x_i \geq 0 \text{ for } i = 1, \ldots, n \]

As the maximization of a function \( z \) is equivalent to the minimization of \( z \) symmetric function, just sometimes treat only one case.

We present some examples of linear programming models formulation, details of which can be found at [Marins]

Determination of production mix Issue:

Formulate a linear programming model to optimize the profit of the daily output of each of three types of brick (S, M, L) produced by a family business, using two types of resources: Hand labor and material, in accordance with the following performance table of technical resources per unit of output.

<table>
<thead>
<tr>
<th>Income</th>
<th>S</th>
<th>M</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hand labour (hr/pc)</td>
<td>7</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Material (Kg/pc)</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

The entire daily output is sold and the profits from the sale of each type of brick, in monetary units ($), is as follows:

<table>
<thead>
<tr>
<th>Sale of profit</th>
<th>S</th>
<th>M</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Income ($)</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

The company has only 300 hours of hand labor and 200 kg of material per day.

Solution:

The aim is to maximize the daily profit!

The daily profit depends on the quantity of bricks of various types produced daily and the daily profit obtained by selling each type of brick!
The decision variables:

\[ x_1 = \text{amount of S type of brick, produced daily} \]

\[ x_2 = \text{number of M type of brick, produced daily} \]

\[ x_3 = \text{amount of L type of brick, produced daily} \]

Maximizing the objective function in monetary units ($):

\[ z = 4x_1 + 2x_2 + 3x_3 \]

Constraints (daily):

\[ 7x_1 + 3x_2 + 6x_3 \leq 300 \text{ hand labor} \]

\[ 4x_1 + 4x_2 + 5x_3 \leq 200 \text{ material} \]

The non-negativity of decision variables (quantity produced is always non-negative):

\[ x_1 \geq 0 \]

\[ x_2 \geq 0 \]

\[ x_3 \geq 0 \]

The linear programming model is formulated as follows.

Later we will see how it is solved by finding the optimal production quantities of different types of brick!

Advertising campaign Problem:

A company of beauty products to perform a publicity campaign in the media: internet, television, radio and newspaper.

A market survey revealed the following technical data on the audience (thousand) and cost ($) advertising in these media:

<table>
<thead>
<tr>
<th></th>
<th>internet</th>
<th>TV</th>
<th>Radio</th>
<th>Newspaper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost per issue ($)</td>
<td>5</td>
<td>40</td>
<td>30</td>
<td>15</td>
</tr>
<tr>
<td>Global audience (Kpes.)</td>
<td>50</td>
<td>400</td>
<td>500</td>
<td>100</td>
</tr>
<tr>
<td>Female audience (Kpes.)</td>
<td>20</td>
<td>250</td>
<td>150</td>
<td>50</td>
</tr>
</tbody>
</table>

The company wants to spend up to 800 monetary units ($) and additionally:

- reaching 2,000 women, make a minimum of 5 placements on TV between 10 and 20 placements radio, make between 5-10 placements in newspapers

Formulate a linear programming model to optimize the hype of the company.
Solution:

The aim is to reach the largest audience possible.

The audience depends on the number of placements for each media type.

The decision variables:

\[ x_1 \] = number of placements on the Internet
\[ x_2 \] = number of placements on TV
\[ x_3 \] = number of placements in radio
\[ x_4 \] = number of placements in the newspaper

Maximization objective function in the number of people affected (Kpes.):

\[ z = 50x_1 + 400x_2 + 500x_3 + 100x_4 \]

with constraints:

\[ 5x_1 + 40x_2 + 30x_3 + 15x_4 \leq 800 \] the total cost of the campaign
\[ 20x_1 + 250x_2 + 150x_3 + 50x_4 \geq 2000 \] the female audience
\[ x_2 \geq 5 \] of placements on TV
\[ 10 \leq x_3 \leq 20 \] of placements in radio
\[ 5 \leq x_4 \leq 10 \] of placements in the newspaper

and the non-negativity of the decision variables (number of placements is always non-negative):

\[ x_1 \geq 0 \]
\[ x_3 \geq 0 \]
\[ x_2 \geq 0 \]
\[ x_4 \geq 0 \]

The linear programming model is formulated as shown above.

Staff training Problem [Marins]:

An assembly company has a training program at work to the production line operators.

Operators already trained can work as trainers in this program and were responsible for 10 trainees each.

After the training for one month, the company plans to hire up to end of the production period, only 7 of each class of 10 students.

The company currently has 130 contracted operators.

To fulfill its order book the company needs for the coming months, on the production line: 100 operators in January, 150 in February, 200 in March and 250 in April.

The production period ends at the end of April.

He monthly salary ($ in monetary units) of workers and students is as follows:
The union does not allow dismissal of operators before the contract expires.

Find a linear programming model that provides a minimal cost of training program and meets the company’s requirements in terms of the number of operators available in each month.

Solution:

The objective is to meet the number of operators in the production line, with minimum labor cost.

The monthly salary cost is determined by the number of hired workers and the number of trainees.

The decision variables are:

\[ x_{ij} \] is the number of people in the salary category during the month

In full:

\[ x_{11} \] = number of workers on the production line in January
\[ x_{21} \] = number of trainers workers in January
\[ x_{31} \] = number of idle workers in January
\[ x_{41} \] = number of students in January
\[ x_{12} \] = number of workers on the production line in February
\[ x_{22} \] = number of trainers workers in February
\[ x_{32} \] = number of idle workers in February
\[ x_{42} \] = number of trainees in February
\[ x_{13} \] = number of workers on the production line in March
\[ x_{23} \] = number of trainers workers in March
\[ x_{33} \] = number of idle workers in March
\[ x_{43} \] = number of trainees in March
\[ x_{14} \] = number of workers on the production line in April
\[ x_{24} \] = number of trainers workers in April
\[ x_{34} \] = number of idle workers in April
\[ x_{44} \] = number of trainees in April
Minimization of the objective function in monetary units ($):

\[
\sum_{j=1}^{4} \left( 700x_{1j} + 800x_{2j} + 500x_{3j} + 400x_{4j} \right)
\]

with constraints:

decision variables with fixed value (are constant):

\[
x_{11} = 100 \text{ workers in production in January}
\]

\[
x_{12} = \text{workers in production in February}
\]

\[
x_{13} = \text{workers in production in March}
\]

\[
x_{14} = \text{workers on the production line in April}
\]

variables related to each other:

\[
x_{41} = x_{21} \text{ relationship trainees - trainer in January}
\]

\[
x_{42} = x_{22} \text{ relationship trainees - trainer in February}
\]

\[
x_{43} = x_{23} \text{ relationship trainees - trainer in March}
\]

\[
x_{44} = 0 \text{ April no graduates needed}
\]

\[
x_{24} = 0 \text{ April no trainers needed}
\]

The total number of operators at the beginning of each month equals the number of operators in production plus the number of operators more trainers the number of idle operators:

\[
130 = x_{11} + x_{21} + x_{31}
\]

\[
130 + 0.7x_{41} = x_{12} + x_{22} + x_{32}
\]

\[
130 + 0.7x_{41} + 0.7x_{42} = x_{13} + x_{23} + x_{33}
\]

\[
130 + 0.7x_{41} + 0.7x_{42} + 0.7x_{43} = x_{14} + x_{24} + x_{34}
\]

The non-negativity of the decision variables (number of operators or graduates is always non-negative):

\[
x_{ij} \geq 0 \text{ for any } i, j = 1,2,3,4
\]

The linear programming model is formulated as shown above.

Mix in chemical industry Problem [Marins]:

A chemical industrial company, two products, A and B, are made from two chemical operations, 1 and 2.

Each unit product A requires 2 hours of operation 1 and operation 2 3 hours.

Each product B unit requires 3 hours of operation 1 and operation 2 4 hours.
In the production of each product unit B gets up 2 units of product C as a byproduct and at no extra cost.

Up to 5 C product units will be sold at a profit, but the rest must be destroyed.

It is known that:

The product generates a $4 profit per unit.

Product B generates a $10 profit per unit.

Product C generates a $3 per unit profit if sold.

Product C generates a cost of $2 per unit if it is destroyed

The total time available to perform the first operation is 16 hours and the total time for operation is 2 to 24 hours.

Determine a linear programming model to maximize the company’s profit.

Solution:

Note that the profit from the sale of product A and B is a linear function [Marins]:

With respect to the product C that does not occur [Marins]:

The goal is to maximize profit.

The gain depends on the quantity of product A, B, C produced and the amount of C product also destroyed.

The decision variables

\[ x_1 = \text{amount of product A produced} \]

\[ x_2 = \text{amount of product B produced} \]

\[ x_3 = \text{amount of product C produced} \]

\[ x_4 = \text{amount of product C destroyed} \]

Maximize the objective function:

\[ z = 4x_1 + 10x_2 + 3x_3 - 2x_4 \]

with constraints:

\[ 2x_1 + 3x_2 \leq 16 \quad \text{time available for operation 1} \]

\[ 3x_1 + 4x_2 \leq 24 \quad \text{time available for operation 2} \]

\[ 2x_2 = x_3 + x_4 \quad \text{each 1 pc. B result 2 pcs. C} \]

\[ x_3 \leq 5 \quad \text{up to 5 units of C can be sold} \]

and the non-negativity of the decision variables (quantity produced is always non-negative):

\[ x_i \geq 0 \text{ for any } i = 1,2,3,4 \]
The linear programming model is formulated as follows

Load balancing machine use Problem [Marins]

A machine shop work 8 hours a day and produces a piece formed with two components.

Each component is made with a drill and a milling cutter, and the productivity of each machine in the manufacturing of the part is as follows (in minutes):

<table>
<thead>
<tr>
<th>Component</th>
<th>Drilling machine</th>
<th>Milling cutter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>15</td>
</tr>
</tbody>
</table>

The company has first milling drill and 5 cutters, but is intended to balance the load of these machines, so that the difference of time of use between the two does not exceed 30 minutes per day.

Find a linear programming model that maximizes the production of parts (complete sets), with balancing machines.

Solution:

The objective is to maximize the number of parts produced.

The number of pieces depends on the smallest number of manufactured components.

Decision variables:

\[ x_1 \] number of manufactured components 1
\[ x_2 \] number of manufactured components 2

The number of parts (set of two components) is determined by the minimum between the number of manufactured components of the same type.

The function \[ z = \min(x_1, x_2) \] is not linear but can be linearized using a new variable \[ y = \min(x_1, x_2) \]

Maximize the objective function:

\[ z = y \text{ with constraints} \]

\[ y \leq x_1 \] num. parts does not exceed num. Component
\[ y \leq x_2 \] num. parts does not exceed num. Component

\[ 3x_1 + 5x_2 \leq 480 \quad \text{8 hour (480 min), 1 drill} \]
\[ 20x_1 + 15x_2 \leq 480 \times 5 \quad \text{8 hour (480 min), 5 cutters} \]

\[ |(3x_1 + 5x_2) - (4x_1 + 3x_2)| \leq 30 \text{load balancing machines} \]
and the positivity of the decision variables (quantity produced is always non-negative):

\[ x_i \geq 0 \text{ for any } i = 1, 2 \]

The linear programming model is formulated as follows

Team Design Problem:

A factory inspectors want to determine how to allocate a given quality control task.

The information gathered by the team of Operational Research during the working visit to the Factory are:

- The market offers 8 inspectors Level 1 earning $4 per hour, which can check the parts at a rate of 25 pieces per hour with an accuracy (accuracy) of 98%.
- The market has 10 inspectors Level 2 earning $3 per hour, which can check the parts at a rate of 15 pieces per hour with an accuracy of 95%.
- The factory wants at least 1800 parts to be inspected during an 8-hour working day.
- Each mistake made by inspectors in controlling the quality of the parts causes damage to the factory by $2 per poorly inspected piece.

Formulate a linear programming model to optimize the cost of daily inspection of the factory.

Solution:

The objective is to minimize the cost of inspection

The cost of inspection depends on the number of inspectors level 1 and level 2 and mistakes made by them during the inspections.

Decision variables:

- \( x_1 \) = number of inspectors level 1
- \( x_2 \) = number of inspectors level 2

Minimize the objective function (8-hour working day):

\[ z = 4(8)x_1 + 3(8)x_2 + 2(15(8)(0.05)x_1 + 25(8)(0.02)x_2 \]

Simplifying \( z = 40x_1 + 36x_2 \)

With constraints:

- \( x_1 \leq 8 \) constraint labor market
- \( x_2 \leq 10 \) constraint labor market
- \( 25(8)x_1 + 15(8)x_2 \geq 1800 \) inspection requirement

and the non-negativity of the decision variables (number of inspectors is always non-negative):

\( x_i \geq 0 \text{ for any } i = 1, 2 \)
The linear programming model is formulated as follows

**Conclusion**

The scientific decision-making requires the construction of a decision model that matches the reality on which the decision should focus. As the decision variables involved in the model can be numerous, it is convenient that the model formulated is to be solved with the help of computational mathematics

**Assignment**

Problem # 1

A family company produces two types of brick (S, M) using two types of resources: Hand labor and material, in accordance with the following table of resources per unit of output produced technique.

<table>
<thead>
<tr>
<th>Income</th>
<th>S</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hand labour (hr/pc)</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>Material (Kg/pc)</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

The entire daily output is sold and the profits from the sale of each type of brick, in monetary units ($), is as follows:

<table>
<thead>
<tr>
<th>Profit without sale</th>
<th>S</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Income</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

The company only has 200 hours of hand labor and 150 kg of material per day.

Formulate a linear programming model for this problem in order to maximize the daily profit.

Problem # 2

A factory produces two products P1 and P2 in 3 machines M1, M2 and M3, indicating in the table below the processing times (in hours) and net income (in money units) from the sale of items.

<table>
<thead>
<tr>
<th>Article</th>
<th>Laundry. M1</th>
<th>Laundry. M2</th>
<th>Laundry. M3</th>
<th>Profit</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0.25 hr</td>
<td>0.40 hr</td>
<td>0 hr</td>
<td>2</td>
</tr>
<tr>
<td>P2</td>
<td>0.50 hr</td>
<td>0.20 hr</td>
<td>0.80 hr</td>
<td>3</td>
</tr>
</tbody>
</table>

For a time of preparation of 40 hours per week, formulate a linear programming model that will maximize the profit of the factory.

Problem # 3

A textile factory labors in three shifts of eight hours each: 1st round from 7 am to 15 pm, 2nd shift from 15 to 23 hours and 3rd round of 23 hours to 7 hours.
Each shift requires pattern makers, seamstresses and packers who earn an hourly wage, respectively 23, 19:08 currencies ($).

The designers and seamstresses earn an additional 2 A / hour and the packaging an additional 1 one / hour when working on the last of the indicated shifts.

Production needs require, at every turn, 1 hour modeler for every hour 3 hours seamstress, may not be more than 200 hours of sill at every turn.

It is intended that the total hours of designers and seamstresses work is at least 400 hours in the morning shift, 376 hours in the afternoon and 270 hours on the night shift.

During each turn one must perform at least 600 hours of work.

Formulate a linear programming model for the given problem.

SOLUTION:

1. Formulation:

Decision variables:

\[ x_1 \]: Amount of brick S
\[ x_2 \]: Amount of brick M

Maximizing the objective function in monetary units ($):

\[ z = 4x_1 + 2x_2 \]

subject to constraints (daily):

\[ 7x_1 + 3x_2 \leq 200 \]
\[ 4x_1 + 4x_2 \leq 150 \]

Non-negativity:

\[ x_1 \geq 0 \]
\[ x_2 \geq 0 \]

2. Formulation

Decision variables

\[ x_{11} \]: number of modelers in the 1st round
\[ x_{21} \]: number of seamstresses in the 1st round
\[ x_{31} \]: number of packagers in the 1st round
\[ x_{12} \]: number of modelers in the 2nd round
Unit 1. Linear Programming, the Simplex Method

Minimization of the objective function (cost per day):

\[
\begin{align*}
    z &= 23(x_{11} + 8x_{12}) + 19(8x_{21} + 8x_{22}) \\
    &\quad + 8(8x_{31} + 8x_{32}) + 25(8)x_{13} + 21(8)x_{23} + 9(8)x_{33}
\end{align*}
\]

subject to the constraints (daily):

\[
\begin{align*}
    3x_{11} &= x_{21} & \text{reason seamstress-modeler in the 1st round} \\
    3x_{12} &= x_{22} & \text{reason seamstress-modeler in the 1st round} \\
    3x_{13} &= x_{23} & \text{reason seamstress-modeler in the 1st round} \\
    8x_{31} &\leq 200 & \text{packagers per shift} \\
    8x_{32} &\leq 200 & \text{packagers per shift} \\
    8x_{33} &\leq 200 & \text{packagers per shift} \\
    8x_{11} + 8x_{21} &\geq 400 & \text{minimum number of hours for designers and seamstresses} \\
    8x_{12} + 8x_{22} &\geq 376 & \text{minimum number of hours for designers and seamstresses} \\
    8x_{13} + 8x_{23} &\geq 270 & \text{minimum number of hours for designers and seamstresses} \\
    8x_{11} + 8x_{31} + 8x_{32} &\geq 600 & \text{minimum number of hours per shift} \\
    8x_{12} + 8x_{22} + 8x_{32} &\geq 600 & \text{minimum number of hours per shift} \\
    8x_{13} + 8x_{23} + 8x_{33} &\geq 600 & \text{minimum number of hours per shift}
\end{align*}
\]

Non-negativity constraints

\[
    x_{ij} \geq 0, \quad i, j = 1, 2, 3
\]

Activity 2 The Simplex table method

Introduction

The simplex method, a general procedure for solving linear programming problems.

Developed by the brilliant George Dantzig in 1947, it has proved to be a remarkably efficient method that is used routinely to solve huge problems on today’s computers.

In this activity, the different features of the algorithm are presented. The simplex method is an algebraic procedure. However, its underlying concepts are geometric. Therefore before getting into any algebraic
Activity Details

Graphical method

The resolution method of a linear programming problem with a graphical method are illustrated in this example.

Consider Problem # 1 in the assessment of Activity 1.1

The formulation of this problem is given by:

Maximizing the objective function in monetary units ($) (z):

\[ z = 4x_1 + 2x_2 \]

Constraints (daily):

\[ 7x_1 + 3x_2 \leq 200 \]
\[ 4x_1 + 4x_2 \leq 150 \]

Non-negativity constraints:

\[ x_1 \geq 0 \]
\[ x_2 \geq 0 \]

Restrictions and non-negativity of decision variables define the domain of the objective function:

The non-negativity limits the company’s production \((x_1, x_2)\) to the first quadrant.

The first constraint \(7x_1 + 3x_2 \leq 200\) limits production of the company in the region below the line defined by \(7x_1 + 3x_2 = 200\).

The second constraint \(4x_1 + 4x_2 \leq 150\) limits the production to the region below the line defined by \(4x_1 + 4x_2 = 150\)

The intersection of these regions is the feasible region:

As the objective function \(z(x_1, x_2)\) is linear, the extreme values can only be located on the edge of this region, unless \(z\) is constant (in which case any point of the feasible region is great).

Moreover, if \(z\) is not constant, extreme values can only occur on board the vertices of the feasible region, unless one of the edges coincide with the end of the objective function and in this case any point of the edge is great.

Thus, the candidates for the basic feasible solution are the pairs \((x_1, x_2)\) at the corners, where you can calculate the values of the objective function \(z\)
Unit 1. Linear Programming, the Simplex Method

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( z )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>175/8</td>
<td>125/8</td>
<td>475/4</td>
</tr>
<tr>
<td>0</td>
<td>150/4</td>
<td>75</td>
</tr>
<tr>
<td>200/7</td>
<td>0</td>
<td>800/7</td>
</tr>
</tbody>
</table>

The optimum operating company is \( x_1 = 175/8, x_2 = 125/8 \) where the sales income reaches the maximum, the optimal solution of \( z = 475/4 \) (118.75).

Firstly there is a problem: the blocks are produced in whole units and not in fractions.

For exact integer solutions, uses the method of the entire optimization but for this course, the figure rounded to a whole less is sufficient:

Operating optimal point: \( x_1 = 21 \text{ wn}, x_2 = 15 \text{ wn} \)

Great solution: (getting even some stock in the warehouse)

The graphical method presents a good intuition about the problem of optimization when there are a maximum of three (3) decision variables.

With four or more decision variables, one cannot use the graphical method but can use the algebraic method, following the script left by intuitive graphical method.

Algebraic method

The algebraic method is to solve a system of equations, but in most cases, and positivity constraints are inequalities and not equations. To transform inequalities into equations new variables, called slack variables are introduced.

Problem # 1 For example Activity 1.1 in the Assessment:

\[
7x_1 + 3x_2 \leq 200 \quad \text{s transformed into} \quad 7x_1 + 3x_2 + s_1 = 200 \quad \text{where} \quad s_1 \geq 0
\]

\[
4x_1 + 4x_2 \leq 150 \quad \text{s transformed into} \quad 4x_1 + 4x_2 + s_2 = 150 \quad \text{where} \quad s_2 \geq 0
\]

The two slack variables \( s_1, s_2 \) are non-negative and can be interpreted as warehouse stock.

The objective of profit maximization necessarily requires that this existence is consumed in production.
It is necessary to solve a system of \( m \) equations (\( m = 2 \)) and \( n \) variables (\( n = 4 \)) (including \( d \)-decision variables and \( n-d \) slack variables)

\[
\begin{align*}
7x_1 + 3x_2 + s_1 &= 200 \\
4x_1 + 4x_2 + s_2 &= 150
\end{align*}
\]

Max, \( m-n = 2 \) variables are free (non-basic) and the two others are dependent (basic).

In general the free variables can take any value, and in the algebraic method the free variables are set to zero.

The possibilities to choose 2 of the 4 variables and set them to zero are given by the equations:

1 - Setting the variables to zero \( x_1 = 0, x_2 = 0 \) we obtain two equations:

\[
\begin{align*}
7(0) + 3(0) + s_1 &= 200 \\
4(0) + 4(0) + s_2 &= 150
\end{align*}
\]

with solutions \( (x_1, x_2, s_1, s_2) = (0, 0, 200, 150) \)

2 - Setting to zero the variables \( x_1 = 0, s_1 = 0 \) we obtain two equations:

\[
\begin{align*}
7(0) + 3x_2 + (0) &= 200 \\
4(0) + 4x_2 + s_2 &= 150
\end{align*}
\]

with solutions \( (x_1, x_2, s_1, s_2) = (200/3, 0, -350/3) \) which is impossible because it violates the non-negativity constraints.

3 - Setting to zero the variables \( x_1 = 0, s_2 = 0 \) we obtain two equations:

\[
\begin{align*}
7(0) + 3x_2 + s_1 &= 200 \\
4(0) + 4x_2 + (0) &= 150
\end{align*}
\]

with solutions \( (x_1, x_2, s_1, s_2) = (150/4, 175/2, 0) \)

4 - Setting to zero the variables \( x_2 = 0, s_1 = 0 \) we obtain two equations:

\[
\begin{align*}
7x_1 + 3(0) + (0) &= 200 \\
4x_1 + 4(0) + s_2 &= 150
\end{align*}
\]

with solutions \( (x_1, x_2, s_1, s_2) = (200/7, 0, 0, 250/7) \)

5 - Setting to zero the variables \( x_2 = 0, s_2 = 0 \) we obtain two equations:

\[
\begin{align*}
7x_1 + 3(0) + s_1 &= 200 \\
4x_1 + 4(0) + (0) &= 150
\end{align*}
\]
with solutions \((x_1, x_2, s_1, s_2) = (150/4, 0, -125/2, 0)\) which is impossible because it violates the non-negativity constraints.

6. Setting to zero the variables \(s_1 = 0, s_2 = 0\) we obtain two equations:

\[
7x_1 + 3x_2 + (0) = 200
\]

\[
4x_1 + 4x_2 + (0) = 150
\]

with solutions \((x_1, x_2, s_1, s_2) = (175/8, 125/8, 0, 0)\)

The following table is a summary of the results:

<table>
<thead>
<tr>
<th>Pt</th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(s_1)</th>
<th>(s_2)</th>
<th>(z_j)</th>
<th>Base</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>200</td>
<td>150</td>
<td></td>
<td>(s_1, s_2)</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>200/3</td>
<td>0</td>
<td>-350/3</td>
<td>400/3</td>
<td>(x_2, s_2)</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>150/4</td>
<td>175/2</td>
<td>0</td>
<td>150/2</td>
<td>(x_2, s_1)</td>
</tr>
<tr>
<td>4</td>
<td>200/7</td>
<td>0</td>
<td>0</td>
<td>250/7</td>
<td>800/7</td>
<td>(x_1, s_2)</td>
</tr>
<tr>
<td>5</td>
<td>150/4</td>
<td>0</td>
<td>-125/2</td>
<td>0</td>
<td>150</td>
<td>(x_1, s_1)</td>
</tr>
<tr>
<td>6</td>
<td>175/8</td>
<td>125/8</td>
<td>0</td>
<td>0</td>
<td>475/4</td>
<td>(x_1, x_2)</td>
</tr>
</tbody>
</table>

Graphically, the points correspond to the intersections of the lines that define the regions of interest:

The solutions (or points) are called basic feasible solutions if the points are on the edge of the feasible region, if they are outside the feasible region the basic solutions are infeasible.

At least one of the feasible solutions is the optimal solution.

If two solutions are then satisfying all points on the edge connecting these two solutions are also great.

**Activity 3 Simplex Table Algorithm**

The algebraic method “waste time” to calculate all the basic solutions including the infeasible ones.

You can “save time” if it is possible to develop an algorithm that:
1. Do not calculate the infeasible solutions

2. Find an improvement during each iteration in the feasible solution

3. Quickly identify the optimal solution when this is achieved

In 1948, George Dantzig created the Simplex algorithm that initializes a basic point and iterates to a basic point which improves the solution, and ends when there is a better solution.

The simplex method can be arranged in tabular form, as a linear programming problem formulated in $\mathbf{m}$ equations in $\mathbf{n}$ variables (including $\mathbf{d}$ decision variables and $\mathbf{n-d}$ slack variables)

\[
z = \sum_{j=1}^{d} c_j x_j + \sum_{j=d+1}^{n} 0 s_j \quad \text{(slack variable not contributing)}
\]

\[A \mathbf{x} \leq \mathbf{b} \quad \text{where} \quad i = 1, \ldots, m
\]

For example, Problem #1 in the Activity 1.1 Assessment, the formulation is given by:

Maximizing the objective function in monetary units ($\$$):

\[z = 4x_1 + 2x_2 + 0s_1 + 0s_2\]

Constraints (daily):

\[7x_1 + 3x_2 + s_1 = 200\]

\[4x_1 + 4x_2 + s_2 = 150\]

Non-negativity constraint:

\[7x_1 + 3x_2 + s_1 = 200\]

\[4x_1 + 4x_2 + s_2 = 150\]

Startup:

\[
\begin{array}{cccccc}
\mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \mathbf{c}_4 & 0 \\
\hline
\mathbf{x}_1 & \mathbf{x}_2 & \mathbf{s}_1 & \mathbf{s}_2 & \text{RS} & \theta \\
0s_1 & 7 & 3 & 1 & 0 & 200 \\
0s_2 & 4 & 4 & 0 & 1 & 150 \\
\end{array}
\]

The column headings are the $\mathbf{m}$ variables (including the slack variable), the right side of the constraints's equation (RS) and a $\theta$ column which is the maximum amount the variable in the base can have without leaving the feasible region and therefore decides on the next iteration.

In line above the header are placed the coefficients that appear in the objective function with each variable.
In each row below there is the equation that calculates each variable in the base.

The initialization is done with the slack variables, in the base and therefore the corresponding columns appear the columns of the identity matrix (yellow highlighting).

Starting the simplex table is completed.

1st Iteration

<table>
<thead>
<tr>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
<th>$0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$x_2$</td>
<td>$s_1$</td>
<td>$s_2$</td>
<td>RS</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

A new line $c_j^T \cdot z_j$ calculates the difference between $c_j$ of each column with the inner product of the vector of coefficients of the variables based on the coefficient $a_{i j}$ of each of the columns for example the first column (green highlight):

$$4 - (0 \cdot 7 + 0 \cdot 4) = 4$$

The other columns:

$$2 - (0 \cdot 3 + 0 \cdot 4) = 2$$

$$0 - (0 \cdot 1 + 0 \cdot 0) = 0$$

$$0 - (0 \cdot 0 + 0 \cdot 1) = 0$$

and mark the largest value as the variable $x_1$ corresponding to this column, enters the base. (If all the calculated values are zero or negative then the algorithm ends)

Along the same line and in the RS column is calculated $z_j = 0 \cdot 200 + 0 \cdot 0 = 0$ a result that matches the value of $z = 0$ for the decision variables leaving the base $x_1$.

$x_1 = 0, x_2 = 0$.

To know which variable leaves the basis calculate the column dividing the Right Side (RS) by the $a_{i1}$ coefficients of the corresponding column to the variable $x_1$ entering the base $\theta_1 = \frac{200}{7}$ (smaller of the two)

$$\theta_2 = \frac{150}{4}$$
Note that these values correspond to $x_2 = 0$ and alternatively $s_1 = 0$ and $s_2 = 0$.

The variable $(s_1)$, which corresponds to the line of the lowest of these values, out of the base.

The coefficient $a_{11} = 7$ corresponding to the column of the variable enteind and the variable in the line at the left is called pivot and will be used to perform the process of Gaussian elimination on lines where: divides the whole line by the coefficient $a_{11} = 7$

uses $L_{1}$ line to eliminate all other factors in that column, with line operations, starting with $L_{2} - 4L_{1}$:

$$4 - 4(7/7) = 0$$
$$4 - 4(3/7) = 16/7$$
$$0 - 4(1/7) = -4/7$$
$$1 - 4(0/7) = 1$$
$$150 - 4(200/7) = 250/7$$

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th>RS</th>
<th>$\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$4x_1$</td>
<td>1</td>
<td>3/7</td>
<td>1/7</td>
<td>0</td>
<td>200/7</td>
<td></td>
</tr>
<tr>
<td>$0s_2$</td>
<td>0</td>
<td>16/7</td>
<td>-4/7</td>
<td>1</td>
<td>250/7</td>
<td></td>
</tr>
</tbody>
</table>

It appears that in the column of the variables that underlie appear the columns of the identity matrix.

2nd Iteration:
Repeat the steps of the previous iteration:

<table>
<thead>
<tr>
<th>(c_1 = )</th>
<th>(c_2 = )</th>
<th>(c_3 = )</th>
<th>(c_4 = )</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>(x_2)</td>
<td>(s_1)</td>
<td>(s_2)</td>
<td>RS</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(c_{j-Z})</td>
<td>(4)</td>
<td>(2)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>3/7</td>
<td>1/7</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>16/7</td>
<td>-4/7</td>
<td>1</td>
</tr>
<tr>
<td>(c_{j-Z})</td>
<td>(0)</td>
<td>(2/7)</td>
<td>-4/7</td>
<td>0</td>
</tr>
</tbody>
</table>

Calculating \(c_{j-Z}\) verifies that the largest positive value corresponding to column \(x_2\) entering.

The value of 800/7 corresponds to the increase of \(z\) during the iteration.

The variable \(s_2\) leaves the base.

The end:

The following iteration finds that the values of \(c_{j-Z}\) is zero or negative, then the iteration ends and the optimum value is \(z = 475/4\) obtained in points \(x_1 = 175/8\) e \(x_2 = 125/8\).
Graphically we can see that this is a variant of the method “gradient”, restricted the edge of the feasible region:

- From an initial feasible point, the Simplex table algorithm chooses the path of faster increment in the value of the objective function (gradient) in the iteration to the next feasible point.

- Gradients methods, including the method of interior points, may be investigated for example in [Convex Optimization, Stephen Boyd, L Van Den Berghe, Cambridge Press, 2008] but will not be treated here.

Some observations on the Simplex table method:

- A variable that left the base can re-enter, combined with other variables;
- The value of $\theta$ is not calculated if it is negative, it reveals that the variable is already feasible in the region;
- The numbers represented in decimal form (not fractional) may generate rounding errors, especially when it is closer to the value 0;

<table>
<thead>
<tr>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>7</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$x_2$</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$s_1$</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_2$</td>
<td>0</td>
<td>16/7</td>
<td>-4/7</td>
<td>1</td>
</tr>
<tr>
<td>$LD$</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>$\theta$</td>
<td></td>
<td></td>
<td></td>
<td>125/8</td>
</tr>
<tr>
<td>$2x_2$</td>
<td>0</td>
<td>1</td>
<td>-1/4</td>
<td>7/16</td>
</tr>
<tr>
<td>$c_{j-Z}$</td>
<td>0</td>
<td>0</td>
<td>-1/2</td>
<td>-1/8</td>
</tr>
</tbody>
</table>
• The presented Simplex table solves the problem of maximizing the objective function.
• In the case of minimizing the objective function \( z \), \( -z \) maximize the symmetrical objective function, as explained in the following unit.

**Conclusion**

This unit were presented computational foundations of tabular SIMLEX method, from a two-dimensional problem, illustrated with graphic and algebraic method.

The tabular Simplex method was applied in detail to a maximization problem of the objective function.

**Activity 4 SIMPLEX with spreadsheet and SOLVER**

**Introduction**

This activity is to solve alinear programming problem of the type maximization using the Simplex algorithm using Excel spreadsheet / Google sheet;

It also Introduces tthe add-on SOLVER on Excel / Google sheet, and solve a maximization-type problem.

**Activity Details**

Spreadsheet with Simplex table

The table form of the Simplex can be automatically inserted in an Excel spreadsheet, or “Google sheet”. As an example, using the problem-solved in the activity 1.1:

The spreadsheet can be further improved, but both excel as the “Google sheet” have an “Add-on” call SOLVER that can be used to solve such problems.

**How to use the SOLVER**

a. Open a new excel spreadsheet/google sheet
b. On the Menu options, click on Add-ons
c. Click on Get Add-ons a new windows open
d. Then select Solver
e. Follow the instructions and the solver will be added
f. It will be shown on the add-ons menu
g. As an example, if you use it to solve the problem Activity 1.1:

**Prepare the Startup table**

The cell formulas are inbuilt and automatically do the calculations:

For more complex problems, both excel as the “Google sheet” have an “Add-on” call SOLVER that can be used to solve such problems.

Some brief instructions on how to use the SOLVER:
Prepare the startup table

Mix of production shown in a column initially with 1 unit of production for each type;

Resources by type of product show that the resources required to produce one unit of each type of product;

Profit per type of product that shows the profit from the sale of 1 unit of each type of product;

Call the SOLVER from the Add-ons menu

It appears to the right of the spreadsheet the SOLVER menu in which arises:

- Set Objective - here indicates the name of the cell that contains the expression of the objective function \( z \) calculated automatically for 1 unit of each type of product;
- To - here chooses the type of optimization: maximizing, minimizing, or a certain value;
- By Changing - here indicates the range of cells where the values of the decision variables to be changed to optimization;
- Subject To - here indicate the restrictions with the help of the buttons Add / Change / Delete to add, change or delete constraints. For the non-negativity constraint indicates the range of cells with the value of the decision variables and puts greater than or equal to \( (> =) \) to zero. For other restrictions indicate is the range of their cells;
- Solving Method - indicates the method used in the resolution. Here only interested in the Standard LP (linear programming);
- Reset All / Insert Example / Solve / Options - are part of the SOLVER menu and here is regulated the restriction of precision, whole optimization, convergence and other parameters that must be investigated by the student.

As an example, we use the Problem # 1 in the assessment of Activity 1.1

The startup framework:

The result of the optimization performed by SOLVER is shown in red

For more information you should consult the HELP of SOLVER.

Conclusion

This unit dealt with finding solution to a maximization problem using the Simplex algorithm in a spreadsheet Excel / Google sheet;

The Add-on SOLVER was also presented and a problem-type maximization was used to illustrate the concept.
Assignment

Answer the following problem with maximum clarity

Problem # 1

Use the graphical method to solve the first example of the Activity 1.1 - Determination of a mix Production problem:

Formulate a linear programming model to optimize the profit of the daily output of each of three types of brick (S, M, L) produced by a family business, using two types of resources: Hand labor and material, in accordance with the following performance table of technical resources per unit of output.

<table>
<thead>
<tr>
<th>Income</th>
<th>S</th>
<th>M</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hand labour (hr/pc)</td>
<td>7</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Material (Kg/pc)</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

The entire daily output is sold and the profits from the sale of each type of brick, in monetary units ($), is as follows:

<table>
<thead>
<tr>
<th>Sale of profit</th>
<th>S</th>
<th>M</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Income ($)</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

The company has only 300 hours of hand labor and 200 kg of material per day.

Problem # 2

Use the SIMPLEX method (manually) to solve the problem above Problem #1;

Problem # 3

Use a spreadsheet to solve Problem # 1 above;

Problem # 4

Use the SOLVER to solve Problem # 1 above;
Unit Summary

In this unit the concepts of mathematical modeling and formalization of linear programming problems were presented;

It was also presented the terminology and basic elements of linear optimization method;

Computational foundations of the Simplex method were treated from a two-dimensional problem, illustrated with graphic and algebraic method;

The Simplex method was applied to a problem in detail, showing the startup cases, iteration and ending.

A problem-type maximization was solved using the Simplex algorithm in a spreadsheet;

Finally the SOLVER was introduced and used to solve a problem-type maximization.

The test is done under supervision. It lasts for Two hours

Unit Readings and Other Resources

The readings in this unit are to be found at course level readings and other resources.
Unit 2. Duality and Sensitivity Analysis

Unit Introduction

This unit introduces the Simplex algorithm table for the objective function minimization problems by introducing the Big M variants and the two phases;

It also discusses the uses spreadsheet and SOLVER to solve complex problems of linear programming dealing with the question of existence and uniqueness of solutions, the initial conditions, iteration and the end of the Simplex algorithm. It presents the relationship between the primal and the dual of a linear programming problem, also presents the dual Simplex and mixed Simplex. The fundamentals of sensitivity analysis and concepts are illustrated with some examples;

Unit Objectives

After completing this unit, you should be able to:

Use Simplex method to manually resolve minimizing problems and maximizing the objective function applying either a variant of the Big M or the two stages;

Use the SOLVER to solve complex linear programming problems;
determine the existence and uniqueness of solutions and launch conditions, iteration and the end of the simplex algorithm and.

Transform a primal problem for its dual solve both interpret the results;

Build the table SIMPLEX dual and mixed;

Identify the most sensitive variables of a linear programming problem and the impact of the variation in the objective function;

Key words

Bounded: \( R \) region or that standard anywhere is a real number

Big M: An unknown positive value greater than any of the problem of numbers

Negative gap: value whose symmetrical offsets inequality

Auxiliary variables: variables used temporarily in solving a problem

Two phases: Resolution in two phases

Standard way: desirable way to start the SIMPLEX table for a maximization problem
Enter the base: variable to be calculated on the basis of other
Out of the base: variable is set to zero
Great alternative: amount equal to the great value
Primal: the problem in the first formalized way
Dual: one obtained from the primal problem
Transposed: mother with inter-exchanged rows and columns
complementarity clearance: zero-sum product of the dual vector with the primal
Product sum: Command Sheets / Excel for domestic product
Dual failure: the difference between the value of the objective function of the primal and dual
Optimality criterion: equal the value of the objective function of the primal and dual
Sensitivity analysis: impact of various changes in the value of the objective function

Learning Activities

Activity 2.1 - Existence of solutions and cases in the implementation of Simplex

Introduction

This activity presents the conditions of existence and uniqueness of solutions of linear programming problems and the linear programming theorem;

It also presents special cases that arise at the beginning iteration and the end of the Simplex algorithm table.

Activity details

2.1.1 Existence of solutions

The existence of solutions of a linear programming problem depends on the feasible region, the monotony of the objective function (ascending or descending) and if the problem is well-placed or not.

In linear programming problems covered here, the feasible region is always a subset of the positive cone (1st quadrant) and when it is defined is always convex (Simplice) because results from the intersections of semi-spaces (half-planes).
When there is no feasible region (intersection of constraints is an empty set) says the problem is not feasible or misplaced.

When there is no empty region feasible, the problem is said to be feasible, and in this case can be delimited or non-delimited as well as the feasible region it is, or may have no solution.

When the region is feasible, the optimal solution is bounded and occurs at the edges, more precisely in Simplice vertices.

The different cases are summarized in the Fundamental Theorem of Linear Programming:

Any linear programming problem (LPP) is feasible, or is not feasible or is not delimited;

LPP has a viable solution then has basic solution (a vertex of the feasible region);

LPP has a great solution this has to be basic feasible solution;

The demonstration of the theorem is done through the following steps [Luenberger]:

Given a linear programming problem in standard form with a matrix $A$ of restrictions with dimension $(m \times n)$ and $m$ entries, $\max z = c^T x$

$Ax = b$

$x \geq 0$

Or there is a region (not empty) bounded feasible, or a non-bounded feasible region or there is no feasible region (intersection of constraints results in an empty set).

Let $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_n]$ be the column vectors of the matrix $A$. Suppose $x = (x_1, x_2, \ldots, x_n)$ is a viable solution. Then, in accordance with the column of $A$

This solution satisfies:

$x_1 \mathbf{A}_1 + x_2 \mathbf{A}_2 + \ldots + x_n \mathbf{A}_n = b$

Assume that the variables $x_i$ exactly $p$ are greater than zero, and for convenience, these are the first variables $p$. Then we can re-write (1) as (2).

$x_1 \mathbf{A}_1 + x_2 \mathbf{A}_2 + \ldots + x_p \mathbf{A}_p = b(2)$.

Then we have to consider two cases, depending on whether the vectors $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_p]$ are linearly independent (LI) or Linearly dependent (LD).

Case 1: Assume that $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_p]$ are Linearly independent. Then $p \leq m$.

If $p = m$, the solution is a basic solution.

If $p < m$, then we can choose $m - p$ vectors between the $n - p$ other such that the set of $m$ vectors formed by $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_p]$ and $m - p$ are chosen Linearly independent, because the rank of $A$ is $m$. Assigning the value zero to $m - p$ corresponding variables, we obtain a feasible basic solution (degenerate in this case).

Case 2: Suppose $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_p]$. They are LD. Then, by definition of Linear dependence, the non-trivial combination of these vectors is zero.
That is, there are constant $y_1, y_2, \ldots, y_p$ with at least one positive, such that
$$y_1A_1 + y_2A_2 + \cdots + y_pA_p = 0 \tag{3}$$
Multiplying (3) by a scalar $\epsilon$ and subtracting from (2) we obtain
$$(x_1 - \epsilon y_1)A_1 + \cdots + (x_p - \epsilon y_p)A_p = b \tag{4}$$
Expression (4) is valid for any $\epsilon$, and for each $\epsilon$, the restrictions $x_i - \epsilon y_i \geq 0$ correspond to a solution of (4), although the restriction $x_i - \epsilon y_i \geq 0$ can be violated.

Putting $y = (y_1, y_2, \ldots, y_p, 0, \ldots, 0)$ we find that any one of (4) solution is: $x = \epsilon y$

(5) For $\epsilon = 0$, $x$ is the original basic feasible solution. When $\epsilon$ grows from zero, the $x$ components may increase, decrease, or remain the same depending on whether the corresponding values of $\epsilon$ are negative, positive or zero. Since we are assuming that at least one of the values of $y_i$ is positive, at least one of its components decreases when $\epsilon$ increases. Let $\epsilon$ increase until one or more of the components become zero, i.e., consider the following value $\epsilon = \min \{ x_i/y_i : y_i \neq 0 \} \tag{6}$

For this value of $\epsilon$, the solution given by (5) is feasible and has a maximum $p-1$ positive variables. Repeating this process, if necessary, we can eliminate the positive variables until you have a feasible solution with the corresponding columns Linearly Independent, a situation where case 1 applies.

3) Suppose $x = (x_1, x_2, \ldots, x_n)$ is a great feasible solution, as in the proof of part 2) above, suppose there are only $p$ variables, $x_1, x_2, \ldots, x_p$. They are positive. We, again, have two cases:

Case 1: Linear Independence corresponds to the case and is treated exactly the same way as above.

Case 2: Linear Dependence corresponds to the case and is also treated in the same manner with the corresponding case, but in this case we have to show that $\epsilon$, the solution (5) is great.

For this, we know that the value of the objective function for the $x = \epsilon y$ and the solution is:
$$c'x - \epsilon c'y \tag{7}$$
for sufficiently small $\epsilon$. $c'x - \epsilon c'y$ is a viable solution for positive and negative values of $\epsilon$. We can then conclude that $c'y = 0$ because it $c'y \neq 0$, a small amount of $\epsilon$ with an appropriate sign could be found to do (7) less than $c'x$, maintaining viability, which contradicts the assumption that $x$ is a great solution. So $c'x$. Therefore, it obtains a new feasible solution with a smaller number of positive components. The rest of the proof then proceeds as in Part 2). The proof of this theorem shows that the resolution of a linear programming problem is reduced to search on all the basic solutions as it was suggested by the simplex method introduced in the previous unit.

During the execution of the Simplex table abnormal situations arise if there is no solution or there is more than one solution to the given problem.

2.1.2 - Initialization of the simplex table

During initialization of the table of the simplex algorithm, we must take into account that the decision variables may be of three types:

- non-negative: $x_i \geq 0$
Non-positive: \( x_j \leq 0 \)

unrestricted in sign: \( x_j \in (-\infty, \infty) \)

Non-negative variables are preferred during the initialization of the simplex table:

When they appear non-positive variables: \( x_j = -x_k \). It creates a new variable with the symmetrical value \( x_j = -x_k \) and the new non-negative variable \( x_k \geq 0 \) is replaced in the objective function and the constraints.

When they appear as unrestricted variables \( x_j \) in sign, it creates two new variables \( x_k, x_m \) whose difference is equal to the initial variable \( x_j = x_k \cdot x_m \) and new variables are replaced in the objective function and the constraints.

The restrictions, if the right side (RS) is not positive, multiplying all the inequality by (-1).

The Constraints can be of three types:

Equality: Equal left side equal to the right side, \( LS=RS (+) \)

Inequality: \( LS \leq RS (+) \)

Inequality: \( LS \geq RS (+) \)

The problem can only be solved algebraically if an equality as the solution exists on the edge of the feasible region.

In the case of the inequality \( LS \leq RS (+) \), positive slack variables are created which are added to the left side \( LS \) to transform the inequality into equality;

In the case of the inequality \( LS \geq RS (+) \), negative slack variables are created which are subtracted the left side \( LE \) to transform the inequality in equality;

It is important here to distinguish the preferred type of restrictions for cases of maximization and minimization of the objective function:

For maximization: \( LS \leq RS \)

For minimize: \( LS \geq RS \)

Using these preferred types of constraints for each of the types of optimization, assures the existence of solution of the linear programming problem.

2.1.3 - Iteration in the simplex Table

During iteration may arise the following questions:

1 - Draw the variable that enters the base - When two or more positive values on the line \( c_j \cdot Z_j \) are identical one is chosen by any decision method. This shows that the iteration of a vertex decisions for the neighbors are indistinguishable:

However, the implementation of an algorithm is necessary to provide the computer a rule to decide between two identical options, so it is common to opt for the decision variable with higher coefficient in the objective function or highest level.
2 - Draw the variable out of the base

When two or more values of $\theta$ are identical, one is chosen based on any decision method (arbitrary).

This reveals degeneracy, i.e. a point of intersection between more than two lines:

Finally, it is noted that the value of $\theta$ can not be negative in the SIMPLEX table since it is designed to find the maximum of the objective function.

If during any iteration a negative value is assigned to the result, then leave it in indeterminate form and the decision on the variable does not enter the base.

2.1.4 - The end of the SIMPLEX table

As seen in maximizing problems, the conditions for termination of the tabular Simplex algorithm termination are met when However, the termination of the tabular Simplex algorithm may raise three questions:

1 - Great alternative

when more than one combination of basic variables provide the same value of the objective function, the basic variable are repeated without the termination conditions being satisfied;

Graphically, at all the edge the value of z is great but as the tabular Simplex method indicates only the two vertices, then two of the vertices provide great value:

2 - Unbounded feasible region - Cannot be determined, absence of solution;

3 - Infeasibility - auxiliary variable does not leave the base even satisfying the termination conditions, the solution is not feasible in view of the restrictions;

For each of these situations, an example illustrating the problem will be given in the evaluation section.
Conclusion

The fundamental theorem of linear programming states that any linear programming problem (LPP) is feasible, or is non-delimited or is not feasible.

LPP has a viable solution then has basic solution (vertex of the feasible region) and a LPPL has great solution then it has to be basic feasible solution; These cases are manifest in the simplex table as follows:

- More than a solution or great alternative when more than a combination of basic variables provide the same value of the objective function, the basic variables are repeated without the termination conditions are met;
- Non-existence of unbounded solution or feasible region when you can not get a determined value; infeasible solution before the restrictions when some auxiliary variable does not leave the base even satisfying the termination conditions,

Assessment

Solve the following problems with the Simplex table and/or SOLVER, as requested, and discuss whether the problem has one, many or no solution.

Problem #1

Consider the linear programming problem to the following formulation variables which are involved in the three types mentioned in 2.1.2:

\[
\begin{align*}
    \text{max}(z) & \quad -x + 4x_2 + x_3 \leq 75 \\
    x & = x_1 - x_2 + 2x_3 \quad x_1 \geq 0 \\
    x & = x_1 - x_2 + 2x_3 \quad x_2 \leq 0 \\
    2x_1 + 2x_2 + x_3 \leq 35 & \quad x_3 \text{ unrestricted in sign}
\end{align*}
\]

Put the problem in standard form for the startup table of the Simplex, replacing the variables given by non-negative variables.

Problem #2

Consider a linear programming problem using the following formulation illustrating the case of great alternative:

\[
\begin{align*}
    \text{max}(z) & \quad 4x_1 + 3x_2 \\
    x & = 4x_1 + 3x_2 \quad 2x_1 + 3x_2 \leq 8 \\
    3x_1 + 2x_2 \leq 12 \\
    x_1 & \geq 0 \\
    x_2 & \geq 0
\end{align*}
\]
Solve the problem using the Simplex table and discuss the case that appear in no basic variable column the value of $c_j z_j = 0$.

Problem #3

Consider a linear programming problem with the following formulation:

$$\max (z)$$

$$z = 4x_1 + 3x_2$$

$$x_1 - 6x_2 \leq 5$$

$$3x_1 \leq 11$$

$$x_1 \geq 0$$

$$x_2 \geq 0$$

Solve the problem using the Simplex table and discuss why is it not possible to continue iterating.

Activity 2.2 - Big M and the Two Phases

Introduction
The Simplex table was developed in Unit 1 to solve the objective function of a maximization problem in standard form.

But it is not always possible to place a linear programming problem in this standard form.

There are cases where the constraints are inequalities or equalities $=$, and there are minimization problems where the objective function is not always possible to start the Simplex at the origin.

This activity introduced the Simplex variants that allow to solve these problems, including the Big M and the Two Phases that in the first instance achieve a starting point and a second time proceed with Simplex iterations.

Activity Details
The Simplex tale was developed in Unit 1 to solve maximization of the objective function problems, whose standard way of maximizing:

$$\max (z)$$

$$f(x) \leq b$$

$$x_i \geq 0$$

But it is not always possible to put a linear programming problem in this standard form.
There are cases where the constraints are inequalities of the type $\geq$ or equalities $=\$. And most importantly, there are problems in standard form of minimization:

$$\min(z)$$

$$f_i(x) \geq b$$

$$x_i \geq 0$$

It is not always possible to start the Simplex at the origin.

The aim of the Big M method or two phases is to create a point with coordinates $(Ma_1, Ma_2)$ where $M$ is a “very large positive number,” and from there find a basic feasible solution (BFS) where it is possible to initialize the Simplex.

NOTE: The sign (+/-) of $M$ in the objective function depends on the problem being maximization or minimization.

Example:

Solve the minimization problem in standard form:

$$\min(z)$$

$$z = 3x_1 + 4x_2$$

$$2x_1 + 3x_2 \geq 8$$

$$5x_1 + 2x_2 \geq 12$$

$$x_1, x_2 \geq 0$$

Solution: First, the problem of minimizing the objective function $Z$ is transformed into a maximization problem for the transformation $\min(z) = \max(-z)$

$$\max(-z)$$

$$(-z) = -3x_1 - 4x_2$$

The constraints require the introduction of two negative slack variables $s_1, s_2 \geq 0$ to obtain the equations:

$$2x_1 + 3x_2 - s_1 = 8$$

$$5x_1 + 2x_2 - s_2 = 12$$

The negative slack variables do not contribute to the objective function $(-z)$ so bring zero coefficients in the objective function:

$$(-z) = -3x_1 - 4x_2 - 0s_1 - 0s_2$$

As the initialization can not be made at the point $x_1 = 0, x_2 = 0$ because it does not
meet the restrictions, it creates a point with coordinates \((Ma_1, Ma_2)\), where \(M\) is a "very large positive number", with the lowest value possible of the objective function \((z)\) obtained by multiplying the artificial variables by the coordinates of the Big M:

NOTE: The sign of \(M\) in the objective function is negative because it is a maximization of \((-z)\)

If the artificial variables \(a_1, a_2\) are not part of the problem, it cannot also be part of the solution: initially the artificial variables are removed from the base and in a second stage, the SIMPLEX is executed.

\[
(-z) = -3x_1 - 4x_2 - 0s_1 - 0s_2 - Ma_1 - Ma_2
\]

2.2.1 - The Big M

The method of the Big M performs both phases in a single framework:

<table>
<thead>
<tr>
<th></th>
<th>-3</th>
<th>-4</th>
<th>0</th>
<th>0</th>
<th>-M</th>
<th>-M</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>(x_2)</td>
<td>(s_1)</td>
<td>(s_2)</td>
<td>(a_1)</td>
<td>(a_2)</td>
<td>RS</td>
</tr>
<tr>
<td>(-Ma_1)</td>
<td>2</td>
<td>3</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(-Ma_2)</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(c_{j \cdot z})</td>
<td>(7M\cdot3)</td>
<td>(5M\cdot4)</td>
<td>(-M)</td>
<td>(-M)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(-Ma_1)</td>
<td>0</td>
<td>11/5</td>
<td>-1</td>
<td>2/5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(-3x_1)</td>
<td>1</td>
<td>2/5</td>
<td>0</td>
<td>-1/5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(c_{j \cdot z})</td>
<td>0</td>
<td>(11M\cdot14)</td>
<td>(14)</td>
<td>(-M)</td>
<td>(2M\cdot3)</td>
<td>(3)</td>
</tr>
<tr>
<td>(-4x_2)</td>
<td>0</td>
<td>1</td>
<td>-5/4</td>
<td>2/11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(-3x_1)</td>
<td>1</td>
<td>0</td>
<td>2/11</td>
<td>-3/11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(c_{j \cdot z})</td>
<td>0</td>
<td>0</td>
<td>-14/11</td>
<td>-1/11</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The initialization is done taking into account both the slack variables as artificial variables in the header.

The artificial variables begin at the base with the coefficients of \(-M\).

While some variables help the basis, it is not necessary to calculate the artificial variables (gray cells) because the values are not used for the calculations.
The optimal values are:

\[ x_1 = \frac{20}{11} \]
\[ x_2 = \frac{16}{11} \]

\[ \max (z) = -\frac{124}{11} \]

Therefore the minimum value of \( z = \frac{124}{11} \)

### 2.2.2 - The Two Phases method

In the method of the two phases, each phase is performed in a separate table.

Consider the same problem as in the previous example:

\[ \max (-z) \]

\[ (z) = -3x_1 - 4x_2 \]

With both negative slack variables \( s_1, s_2 \geq 0 \) the following equations are obtained:

\[ 2x_1 + 3x_2 - s_1 = 8 \]
\[ 5x_1 + 2x_2 - s_2 = 12 \]

Initially put off all the coefficients of the decision variables to zero, it takes the value of \( \theta \) and executes the SIMPLEX table until the artificial variables \( a_1, a_2 \) leave the base:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>-1</th>
<th>-1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>x_1</td>
<td>x_2</td>
<td>s_1</td>
<td>s_2</td>
<td>a_1</td>
<td>a_2</td>
<td>RS</td>
</tr>
<tr>
<td>-a_1</td>
<td>2</td>
<td>3</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>-a_2</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>c_j-z</td>
<td>7</td>
<td>5</td>
<td>-1</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-a_1</td>
<td>0</td>
<td>11/5</td>
<td>-1</td>
<td>2/5</td>
<td></td>
<td></td>
<td>16/5</td>
</tr>
<tr>
<td>0x_1</td>
<td>1</td>
<td>2/5</td>
<td>0</td>
<td>-1/5</td>
<td></td>
<td></td>
<td>12/5</td>
</tr>
<tr>
<td>c_j-z</td>
<td>0</td>
<td>11/5</td>
<td>-1</td>
<td>2/5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x_2</td>
<td>0</td>
<td>1</td>
<td>-5/4</td>
<td>2/11</td>
<td></td>
<td></td>
<td>16/11</td>
</tr>
<tr>
<td>0x_1</td>
<td>1</td>
<td>0</td>
<td>2/11</td>
<td>-3/11</td>
<td></td>
<td></td>
<td>20/11</td>
</tr>
<tr>
<td>c_j-z</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The first phase ends when all auxiliary variables come out of the base; if they do not come out of the base then the problem has no solution.

In the second stage, the values of the last iteration of the first stage table (shown in green) are entirely copied to the second stage of the framework and one proceeds then to the usual simplex iteration until values becomes zero or negative that indicate the end of the iteration:

\[
\begin{array}{cccccc}
 x_1 & x_2 & s_1 & s_2 & RS & \theta \\
-3 & -4 & 0 & 0 & 0 & 0 \\
-4x_2 & 0 & 1 & -5/4 & 2/11 & 16/11 \\
-3x_1 & 1 & 0 & 2/11 & -3/11 & 20/11 \\
c_j & z & j & 0 & 0 & -124/11 \\
\end{array}
\]

In this example it was not necessary to go beyond the 1st iteration to get the results:

\[
x_1 = 20/11
\]

\[
x_2 = 16/11
\]

\[
max (-z) = -124/11
\]

Therefore the minimum value of \( z \) = 124/11

**Conclusion**

It is not always possible to put a linear programming problem in standard form of the Simplex studied in Unit 1 (maximization) where the constraints are inequalities of the type \( \leq \) or equality \( = \). For problems where the objective function is to be minimized, it is not always possible to initialize the Simplex at the beginning of the problem. The coordinates are transformed into a problem of maximizing through the transformation \( \min (z) = \max (-z) \) and the restrictions on the type \( \leq \) or equalities \( = \) negative slack variables and artificial variables are introduced.

With the method of the Big M or the two stages artificial variables are used to the SIMPLEX and to proceed with normal iterations.

**Assessment**

Solve the following problems using variants of the Big M, 2 phases and SOLVER:

**Problem # 1**

A company of beauty products wants to perform an advertising campaign in the media: Internet, television, radio and newspaper.
A market survey revealed the following technical data about the audience (people) and cost ($) in advertising in these media:

<table>
<thead>
<tr>
<th>Media</th>
<th>Prime TV</th>
<th>TV</th>
<th>Radio</th>
</tr>
</thead>
<tbody>
<tr>
<td>cost per program ($)</td>
<td>40</td>
<td>75</td>
<td>30</td>
</tr>
<tr>
<td>Global Audience (pers.)</td>
<td>400</td>
<td>900</td>
<td>500</td>
</tr>
<tr>
<td>Female audience (pers.)</td>
<td>300</td>
<td>400</td>
<td>200</td>
</tr>
</tbody>
</table>

The company reach a minimum audience of 10,000 people and additionally:
- reach at least 5,000 women
- spend not more than $ 500 with TV
- make a minimum of 3 placements in prime TV
- make a minimum of 2 placements in normal TV
- do 5 to 10 placements in radio

Formulate a linear programming model to minimize the cost of the advertising campaign of the company.

**Problem # 2:**
Put problem # 1 in standard form and solve it using the SIMPLEX method

**Problem # 3:**
Solve the same problem using the big M

**Problem # 4:**
Solve the same problem using the Two phases method

**Problem # 5:**
Solve problem # 1 with SOLVER and compare with the previous results.

**Problem # 6:**
Consider a linear programming problem of the following formulation:
Activity 2.3 - Duality

Introduction

This activity shows the relationship between the primal and the dual of a linear programming problem and provides an economic interpretation of the dual.

It also presents the dual and mixed SIMPLEX and its advantages over the methods of Big M and two phases.

Activity Details - The dual problem

For any vector space, there is a dual scalar vector space consisting of all linear functions. Without getting into the theory of vector spaces and dual spaces, this module aims to present the basic principles of the primal-dual relationship and the geometric interpretation provided by the Euclidean spaces $\mathbb{R}^n$ is enough.

Consider for example the space $\mathbb{R}^2$ where the functional (vector-line) acts by the dot product on the vector-column:

$$
\begin{bmatrix}
3 & 2 \\
-1 & 1
\end{bmatrix}
\begin{bmatrix}
-1 \\
1
\end{bmatrix} = 3(-1) + 2(1) = -1
$$

In this view, the space of all column vectors is dual to the space of all the row vectors.

The transpose of this operation (sum-product or inner product) becomes the vector in a vector-column line (linear function) and vice versa:

$$
\begin{bmatrix}
3 & 2 \\
-1 & 1
\end{bmatrix}^T =
\begin{bmatrix}
-1 \\
1
\end{bmatrix}^T
\begin{bmatrix}
3 & 2 \\
1 & 1
\end{bmatrix} =
\begin{bmatrix}
3 & 2 \\
1 & 1
\end{bmatrix}^T
$$

Some insight about the geometric primal-dual ratio can be obtained in the following figure where the primal problem is to FIND the minimum distance from the point P to the convex area (indicated in red solution):

This problem is related to the dual problem of finding the maximum distance from the point P to any tangent to the convex region:
Any linear programming problem (primal) has an associated dual problem.

More specifically, if the primal is:

Minimize $z = cx$

Restrictions $Ax = b$ and $x \geq 0$

Then the dual is

Maximize $w = yb$

Restriction $yA = c$ and $y \geq 0$

The dual problem is constructed from the primal, with the following rules:

If the primal is the maximization problem then the dual is a minimization problem, and vice versa;

For each constraint of the primal corresponds a decision variable of the dual and vice versa;

The coefficients of the objective function of the primal become the right side (RS) of the constraints of the dual;

The transpose of the matrix of coefficients of the constraints in the primal becomes the matrix of the coefficients of the constraints in the dual;

If the primal is maximized and the restriction of inequality is preferred to maximize $(\leq)$ then the variable is non-negative, if the restriction is an equation, ie equal $(=)$ the corresponding variable is unrestricted in sign and the inequality is $(\geq)$ then the variable is non-positive;

If the primal is minimized and the restriction of inequality is the preferred minimizing $(\leq)$ then the variable is non-negative, if the restriction is an equation, ie equal $(=)$ the corresponding variable is unrestricted in sign and is the type of inequality is $(\geq)$, then the variable is non-positive;

If the primal is maximized and the decision variable is then non-negative inequality corresponding restriction of the dual is the preferred minimizing $(\geq)(\leq)$ if it is unrestricted in sign the dual restriction is equal $(=)$ and is non-positive the corresponding dual constraint is $(\leq)$;

If the primal is minimized and the variable is then non-negative inequality corresponding dual constraint is the preferred maximization $(\leq)$ if it is unrestricted in sign the dual constraint is equality $(=)$ and if it is not positive, the corresponding dual constraint is $(\geq)$;

The rules are summarized in the conversion table between the primal (maximization) and dual (minimizing):
<table>
<thead>
<tr>
<th>Primal</th>
<th>$\leftrightarrow$</th>
<th>Dual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective function $z$</td>
<td>$\text{max}$</td>
<td>$\text{min}$</td>
</tr>
</tbody>
</table>

# constraints $\leftrightarrow$ # decision variables

| $\leq$ (Prefer. de max.) | $\leftrightarrow$ | $\geq 0$ |
| $=$                       | $\leftrightarrow$ | Unrestricted in sign |
| $\geq$                    | $\leftrightarrow$ | $\leq 0$ |

Objective function $z$ $\quad \text{min}$ $\leftrightarrow$ $\text{max}$

<table>
<thead>
<tr>
<th># of constraints</th>
<th>$\leftrightarrow$</th>
<th># decision variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\leq$</td>
<td>$\leftrightarrow$</td>
<td>$\leq 0$</td>
</tr>
<tr>
<td>$=$</td>
<td>$\leftrightarrow$</td>
<td>Unrestricted in sign</td>
</tr>
<tr>
<td>$\geq$ (Prefer. de min.)</td>
<td>$\leftrightarrow$</td>
<td>$\geq 0$</td>
</tr>
</tbody>
</table>

coefficients $\leftrightarrow$ Objective function $\leftrightarrow$ R5 of constraints

R5 of constraints $\leftrightarrow$ Objective function

$A$ of constraints $\leftrightarrow$ $A^\tau$ of constraints

Where $A^\tau$ is the transpose of the matrix coefficients of the constraints.

The designation of primal or dual to a linear programming problem is arbitrary, but from the moment that a problem is given designated primal (usually maximization), then the result from the transformation is the dual, and vice versa.

The relationship between the variables of the primal and dual

Consider a primal with $n$ decision variables and $m$ constraints.
The total \((n + m)\) of the decision variables \((n)\) plus the slack variables \((m)\) of the primal and the total dual decision and slack variables are equal:

\[
\begin{array}{cc}
\text{n decision variables of the primal} & \text{m slack variables of the primal} \\
\text{n slack variables of the dual} & \text{m decision variables of the dual}
\end{array}
\]

The relationship between the primal and the dual translates the following principles and theorems:

1 - The principle of complementarity slackness states that the sum-product (dot product) of the vector of the LPP of the decision variables of the primal \(x = (x_1, \ldots, x_n)\) with the dual slack variables \(t = (t_1, \ldots, t_n)\) plus the sum-product vector of the dual decision variables \(y = (y_1, \ldots, y_m)\) with the slack variables of the primal \((s_1, \ldots, s_m)\) is null:

\[
\sum_{j} x_j t_j + \sum_{i} y_i s_i = 0
\]

2 - The weak duality theorem states that if \(x = (x_1, \ldots, x_n)\) is a viable solution to a primal LPP and \(y = (y_1, \ldots, y_m)\) is a viable solution to the dual LPP, then:

\[
\sum_{j} c_j x_j \leq \sum_{i} b_i y_i
\]

The Inequality is compensated by the designated dual failure (dual gap) and in the case of great workable solutions, the dual failure is zero.

3 - The strong duality theorem states that if \(x^* = (x_1, \ldots, x_n)\) is a great viable solution to a primal LPP and \(y^* = (y_1, \ldots, y_m)\) is a great viable solution to the dual LPP, then:

\[
\sum_{j} c_j x^*_j = \sum_{i} b_i y^*_i
\]

That is to say that the optimal values of the primal and dual are equal.

The reciprocal of the strong duality theorem is also valid.

4 - Theorem of the criterion of optimality states that if the primal and dual have feasible solutions then the dual failure is zero, the optimal values are identical

\[z = a\] and the points where these values are obtained are great workable solutions.

Based on these theorems, it is easy to conclude that:

- If the primal is unbounded the dual is infeasible or unbounded
- If the dual is infeasible the primal is unbounded or infeasible
- The optimal solution is only obtained when the primal and the dual are feasible
Use of Simplex table to solve both the primal and the dual

To understand how the Simplex can be used to solve both the primal and the dual, consider the following problem:

$$\max(z)$$

$$z = 6x_1 + 5x_2$$

$$x_1 + x_2 + s_1 = 5$$

$$3x_1 + 2x_2 + s_2 = 12$$

$$x_1, x_2, s_1, s_2 \geq 0$$

where $$S_1, S_2, S_3$$ are the slack variables of the primal.

The implementation of the Simplex table results to the following table:

<table>
<thead>
<tr>
<th></th>
<th>$$c_1 = 6$$</th>
<th>$$c_2 = 5$$</th>
<th>$$c_3 = 0$$</th>
<th>$$c_4 = 0$$</th>
<th>$$0$$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$$x_1$$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>$$x_2$$</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>12</td>
<td>4</td>
</tr>
</tbody>
</table>

$$c_j - z_j$$

$$6$$

$$5$$

$$0$$

$$0$$

$$0$$

$$0$$

$$0$$

$$0$$

$$-1$$

$$1$$

$$3$$

$$4$$

$$6$$

$$0$$

$$1$$

$$0$$

$$-2$$

$$24$$

$$0$$

$$1$$

$$3$$

$$-1$$

$$3$$

$$0$$

$$0$$

$$-3$$

$$-1$$

$$27$$

$$t_1$$

$$t_2$$

$$y_1$$

$$y_2$$
Based on the principle of the complementarity slackness we know that:

the decision variables of the primal relate to the slack variables of the dual

\[
\begin{align*}
    x_1 & \leftrightarrow t_1 \\
    x_2 & \leftrightarrow t_2
\end{align*}
\]

The decision variables of the dual relate to the slack variables of the primal

\[
\begin{align*}
    y_1 & \leftrightarrow s_1 \\
    y_2 & \leftrightarrow s_2
\end{align*}
\]

such that

\[
\sum_{j} x_j t_j + \sum_{l} y_l s_l = 3(0) + 1(0) + 2(0) + 3(0) = 0
\]

In the Simplex table, the last iteration (line \( c \rightarrow z \)) is the solution of the dual (green), with a negative sign, \( y_1 = 3, y_2 = 1 \), with the failures zero \( t_1 = 0, t_2 = 0 \), while the solution of the primal is given (in cyan) in the last iteration of the RS column \( x_1 = 3, x_2 = 1 \) with zero clearance \( s_1 = 0, t_2 = 0 \) implicitly given (because they are not on the base).

Incidentally, is not only the last iteration, at any step of the iteration the principle of complementarity slackness is valid, for example in the second iteration of the Simplex

<table>
<thead>
<tr>
<th></th>
<th>( c_1 = 6 )</th>
<th>( c_2 = 3 )</th>
<th>( c_3 = 0 )</th>
<th>( c_4 = 0 )</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_1 )</td>
<td>0</td>
<td>1/3</td>
<td>1</td>
<td>-1/3</td>
<td>1</td>
</tr>
<tr>
<td>( 6x_1 )</td>
<td>1</td>
<td>2/3</td>
<td>0</td>
<td>1/3</td>
<td>4</td>
</tr>
<tr>
<td>( c_j \rightarrow z_j )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-2</td>
<td>24</td>
</tr>
</tbody>
</table>

we read in the table the values of the primal variables (in cyan)

\[
\begin{align*}
    s_1 & = 1 \\
    x_1 & = 4 \\
    s_2 & = 0 \text{ (non-basic primal)} \\
    x_2 & = 0 \text{ (non-basic primal)}
\end{align*}
\]
and the dual
\[ t_1 = 0 \]
\[ t_2 = -1 \]
\[ y_1 = 0 \]
\[ y_2 = 2 \]

And in any step of the iteration the complementarity slackness occurs:
\[ \sum_{j}^{n} x_j t_j + \sum_{i}^{m} y_i s_i = 0 \]
\[ 4(0) + 0(-1) + 0(1) + 2(0) = 0 \]

We can see step by step the calculations performed by the Simplex table in the second iteration:

The basic variables of the primal are:
\[ s_1 = 1 \]
\[ x_1 = 4 \]

The non-basic variables of the primal are:
\[ s_2 = 1 \text{ (non-basic)} \]
\[ x_2 = 4 \text{ (non-basic)} \]

the corresponding unknown basic of the dual are:
\[ y_2 \]
\[ t_2 \]

and corresponding non-basic of the dual are:
\[ y_1 = 0 \]
\[ t_1 = 0 \]

To calculate the unknowns of the primal, the SIMPLEX uses the restriction of the dual:
\[ y_1 + 3y_2 - t_1 = 6 \Rightarrow 0 + 3y_2 - (0) = 6 \Rightarrow y_2 = 2 \]
\[ y_1 + 2y_2 - t_2 = 5(0) + 2(2) - t_2 = 5 \Rightarrow t_2 = -1 \]

Note that also in this iteration \( w = 24 = z \) is not the optimal value, which is reflected in the fact that the dual is infeasible because \( t_2 = -1 \) does not satisfy the condition \( t_2 \geq 0 \).

The negativity of the dual variables (the dual infeasibility) reflects the non-optimality of the
primal.

Economic Interpretation of the dual

In the example below:

Increasing the value of the first restriction by a value \( \delta \) results in an increase of the objective function by which give a value of \( y_1 = 1 \)

\[
x_1 + x_2 \geq 5 + \delta
\]

\[
3x_1 + 2x_2 \geq 12
\]

solving simultaneously we obtained

\[
x_1 \geq 2 - 2\delta
\]

\[
x_2 \geq 3 + 3\delta
\]

replacing the objective function we obtain the marginal value of \( z \)

\[
z \geq 6(2 - 2\delta) + 5(3 + 3\delta) = 27 + 3\delta
\]

Increasing the value of the second restriction by a value \( \delta \) results in an increase of the objective function by \( \delta \) which give a value of \( y_1 = 1 \)

Therefore, the value of dual variables can be interpreted as the marginal value of the resulting objective function of the incremental variation of the corresponding resource, since the increment is small enough such as not remove the problem in the neighborhood of the optimal point.

From the producer's point of view, it makes sense to buy 1 unit of first resort for up to 3 currencies as the marginal gain from the objective function justifies this investment.

So, economically, the value of the dual variables can be interpreted as the maximum price which the producer offers to pay for each unit of resources used in the primal.

On the other hand, the dual problem is the resource vendor will not sell for less than the optimal values of the dual.

Hence, for the correct formation of the price: satisfaction of the seller and buyer resources, the optimal value of the primal must be equal to the optimal value of the dual.

**Conclusion**

The dual problem and the rules of its construction from the primal are presented. The relationship between the variables of the primal and dual are identified and illustrated. Several principles and theorems that relate in particular to the principle of complementarity slackness, the theorems of duality and optimality criterion are also discussed.
Activity 2.3 - Sensitivity Analysis

Introduction

This activity identified blocks of matrices which suffer iterated transformations during the Simplex algorithm execution process and the sensitivity analysis for the optimal solution if there is any variation in the following:

The objective function

The coefficient matrix of the restrictions on the right side (RS) or even, if you add more variables or restrictions on the initial problem.

Activity Details

Matrix representation of the Simplex algorithm

Here are identified block matrices which undergo iterated transformations during the process of implementation of the simplex algorithm table.

As an example, consider the simplex tableau calculated in the preceding examples:

<table>
<thead>
<tr>
<th>x_1</th>
<th>x_2</th>
<th>s_1</th>
<th>s_2</th>
<th>RS</th>
<th>θ</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>s_1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>s_2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>c_1 = 6</td>
<td>c_2 = 5</td>
<td>c_3 = 0</td>
<td>c_4 = 0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
It identifies, at the beginning, the following matrices:

The column-vector of the variables in in the primal base \( X_B \)

\[
X_B = \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 5 \\ 12 \end{bmatrix}
\]

identifying \( x_3 = s_1 \) and \( x_4 = s_2 \)

A- row matrix of the variable coefficients in the base \( C_B \)

\[
C_B = \begin{bmatrix} 0 \\ 0 \end{bmatrix}
\]

The matrix of the constraints coefficients \( P \)

\[
P = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 3 & 2 & 0 & 1 \end{bmatrix}
\]

where the j-th column is \( P_j \)

The coefficient matrix of restrictions only variable base \( B \)

\[
B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]

The row vector of the dual variables at the base

\[
y = [y_1, y_2] = [0, 0]
\]

The matrix B is a sub-matrix identity at the beginning, as shown in the figure:
The matrix calculations made during the execution of Simplex are referred to these initial matrices:

\[ \overline{P}_j = B^{-1}P_j \]
\[ z_j = C_B \overline{P}_j = C_B B^{-1}P_j = yP_j \quad \text{onde} \quad y = C_B B^{-1} \]
\[ c_j - z_j = c_j - yP_j \]

Where the bar above means that the result goes to the next iteration.

The variable \( x_k \) enters the base it has a greater value \( c_k - z_k \).

It calculates \( \theta \) by formula

\[ (\overline{P}_j)^T \cdot \theta = X_B \]

The variable with the lowest value of the components, called the min out of the base.

Calculate the value of new variables in the base

Reset to variable \( X_B \) removing the bar over and the process is repeated to calculate in the next iteration.

In numbers, the first iteration in the example given:

Enter the base variable \( x_1 \) with the highest value \( c_1 - z_1 \) (note that this iteration)

\( k = 1 \)

Leaves the base the variable \( s_2 = x_4 \) (entering the base)

To calculate the value of new variables in the base:

Continuing with the second iteration in numbers:

Enters the base the variable \( x_2 \) (in this iteration \( k = 2 \))

Leaves the base the variable \( x_3 = s_1 \) (entering \( x_2 \))

The third iteration:

The termination conditions are satisfied and the same great values are found.

Finally, calculate the value of \( as \):

It is interesting to note that \( B^{-1} \) for all iterations appear in the SIMPLEX Table and when the termination is achieved columns of the identity matrix are displayed in columns on the basis of the variables:

**Sensitivity Analysis**

What is the impact of the variation of data from a linear programming problem solution found?

Obviously the impact can be calculated reformulating the problem with the desired changes and comparing results.
However, this method of ‘brute force’ has a high computation cost.

Doing the calculations and running the Simplex algorithm is it a good option for this information?

Moreover, you can get this information from the optimal solution already obtained?

To analyze these issues in detail, consider an example with the linear programming problem (primal) given by:

$$
\text{max } (z) \\
\begin{align*}
    z &= 4x_1 + 3x_2 + 5x_2 \\
    x_1 &+ 2x_2 + 3x_3 \leq 9 \\
    2x_1 &+ 3x_2 + x_3 \leq 12 \\
    x_1, x_2, x_3 &\geq 0
\end{align*}
$$

Using a simplex table

<table>
<thead>
<tr>
<th></th>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
<th>$c_5$</th>
<th>RS</th>
<th>$\theta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0$</td>
<td>$x_4$</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>$0$</td>
<td>$x_5$</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>$c_j$</td>
<td>$z_j$</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$5$</td>
<td>$x_3$</td>
<td>1/3</td>
<td>2/3</td>
<td>1</td>
<td>1/3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>$0$</td>
<td>$x_5$</td>
<td>5/3</td>
<td>7/3</td>
<td>0</td>
<td>-1/3</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$c_j$</td>
<td>$z_j$</td>
<td>7/3</td>
<td>-1/3</td>
<td>0</td>
<td>-5/3</td>
<td>0</td>
<td>15</td>
</tr>
</tbody>
</table>
A variation of $c_2$ can make the $c_2 \cdot z_2 = -\frac{18}{5}$ a positive value and change the optimal solution.

The change required to make it happen can be calculated:

That is, the optimal value is insensitive to an increment change $\frac{33}{5} - 3 = 6$ units of the coefficient of $c_2$.

2) Coefficient Variation of the basic variable in the objective function

Consider a variation in the coefficient of $x_1$ (could also be $x_3$).

Analyzing only the last iteration:

<table>
<thead>
<tr>
<th></th>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
<th>$c_5$</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$x_2$</td>
<td>$x_3$</td>
<td>$x_4$</td>
<td>$x_5$</td>
<td>RS</td>
<td>$\theta$</td>
</tr>
<tr>
<td>5</td>
<td>$x_3$</td>
<td>0</td>
<td>1/5</td>
<td>1</td>
<td>2/5</td>
<td>-1/5</td>
</tr>
<tr>
<td>$c_1$</td>
<td>$x_1$</td>
<td>1</td>
<td>7/5</td>
<td>0</td>
<td>-1/5</td>
<td>3/5</td>
</tr>
<tr>
<td>$c_{j \cdot z_j}$</td>
<td>0</td>
<td>-18/5</td>
<td>0</td>
<td>-8/5</td>
<td>-7/5</td>
<td>138/5</td>
</tr>
</tbody>
</table>

A variation of $c_1$ can change any of the $c_{j \cdot z_j}$ of the non-basic variables for positive values and change the optimal solution, then all should be verified:

That is, the optimal value is insensitive since $c_1$.

3) Variation of the RS (Right Side of constraints) $c_1 \in [5/3, 10]$.

Consider a variation of the right side with an undetermined entry.

The only impact is on the RS, and can be quantified by extracting $b_1$ from the table:

That is, the obtained optimal value is insensitive since $b_1$ is maintained within the range $b_1 \in [6, 36]$. 


4) Variation in the coefficients of the constraints, on a non-basic variable

Consider a variation in a coefficient matrix of constraints, on a non-basic position, say $x_2$ and a variation given by:

The impact of column of $x_2$ affects $2 \cdot z_2$ and this effect can be calculated:

Since the optimum value does not change the value of $q$ is greater than $3/13$.

5) Adding a new decision variable

It will be profitable to introduce a new type of product without changing the stock in the warehouses (initial RS)?

A new product whose amount is to be determined by $x_6$, has a $c_6 x_6$ contribution to the objective function, with $p_6$ coefficients in the coefficient matrix of constraints.

For this example, assume that $c_6 = 6$ and

It creates a new column of $x_6$

<table>
<thead>
<tr>
<th></th>
<th>$c_1$</th>
<th>$c_2$</th>
<th>$c_3$</th>
<th>$c_4$</th>
<th>$c_5$</th>
<th>$c_6$</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_2$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_6$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$RS$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\theta$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>$x_3$</td>
<td>0</td>
<td>1/5</td>
<td>1</td>
<td>2/5</td>
<td>-1/5</td>
<td>-1/5</td>
</tr>
<tr>
<td>4</td>
<td>$x_1$</td>
<td>1</td>
<td>7/5</td>
<td>0</td>
<td>-1/5</td>
<td>3/5</td>
<td>8/5</td>
</tr>
<tr>
<td>$c_6 \cdot z_6$</td>
<td>0</td>
<td>-18/5</td>
<td>0</td>
<td>-8/5</td>
<td>-7/5</td>
<td>3/5</td>
<td>138/5</td>
</tr>
</tbody>
</table>

which are calculated from the values of $x_6$ as

Since it is positive, the new variable enters the base and the variable $x_1$ leave out the base.

If this value was negative, it would not be advisable to introduce the new product.

To calculate the increase in the objective function value, the iteration is continued the Simplex table:
and get a new optimal solution, where the value of the objective function increased from \(138/5\) to \(237/8\).

6) Adding a constraint

Suppose there is a new resource that has not been properly taken into account in the problem and now we must consider it.
As an example, consider a new constraint:
\[ x_1 + x_2 + x_3 \leq 8 \]
As the optimal solution \( x_1 = 27/5, x_2 = 0, x_3 = 6/5 \) satisfies this constraint:
\[(27/5) + (0) + (6/5) = 33/5 \leq 8\]
then this restriction has no impact on the optimal solution.

Now consider a restriction affecting the optimum solution:
\[ x_1 + x_2 + x_3 \leq 6 \]
How to calculate this impact without redoing the problem?

Drawing a table for the two equations, we can write the basic variables in terms of non-basic
Replacing the new restrictions and simplifying we obtain
Adding a slack variable
Creating a new line for the new equation:

<table>
<thead>
<tr>
<th></th>
<th>( c_1 )</th>
<th>( c_2 )</th>
<th>( c_3 )</th>
<th>( c_4 )</th>
<th>( c_5 )</th>
<th>( c_6 )</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>( x_2 )</td>
<td>( x_3 )</td>
<td>( x_4 )</td>
<td>( x_5 )</td>
<td>( s_1 )</td>
<td></td>
<td>LD</td>
</tr>
<tr>
<td>5</td>
<td>( x_3 )</td>
<td>0</td>
<td>1/5</td>
<td>1</td>
<td>2/5</td>
<td>-1/5</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>( x_3 )</td>
<td>1</td>
<td>7/5</td>
<td>0</td>
<td>-1/5</td>
<td>3/5</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>( s_1 )</td>
<td>0</td>
<td>-3/5</td>
<td>0</td>
<td>-1/5</td>
<td>-2/5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>( c_j \cdot z_j )</td>
<td>0</td>
<td>-18/5</td>
<td>0</td>
<td>-6/5</td>
<td>-7/5</td>
<td>0</td>
</tr>
</tbody>
</table>

The negative value in the RS of the new constraint shows that in fact the great solution existing violates the new constraint.

Continuing with the dual Simplex gives the new optimum solution:
given that $x_1 = 9/2, x_2 = 3/2, x_3 = 3/2$ with $z = 51/2$

7) Coefficients of constraints, on a basic variable

What is the impact of a variation in a coefficient of restrictions, on a basic variable?

Here the best alternative is to solve the new problem because it becomes too complex to try to understand the impact from the original Simplex table.

The reason is simple, when there is a change in a coefficient of restrictions, on a basic variable, the “base” changes and you can not use it to calculate the impact on other variables.
**Unit Summary**

The Simplex tableau solve a maximization problem where the right (RS) of the constraints is positive.

To solve a minimization problem, this is transformed into a problem of maximizing the transformation and constraints with negative slack variables and artificial variables are introduced. The Big M or the two stages method are illustrated with some examples.

**Unit Assessment**

Check your understanding!

**Final Examination**

Instructions

Answer all the questions with maximum clarity.

The test is to be done under supervision. The duration is Three (3) hours

**Evaluation Criteria**

Problem # 1

Consider the following problem

Max \( 3x_1 + 2x_2 \)

Subject to

\[ 3x_1 + x_2 \leq 12 \]

\[ x_1 + x_2 \leq 6 \]

\[ 5x_1 + 3x_2 \leq 27 \]

\[ x_1, x_2 \geq 0 \]

a. Solve the problem by the original simplex method and identify the complementary basic solution for the dual problem at each iteration.

b. Write the dual problem, and then solve the problem by the dual simplex method. Compare the resulting sequence of basic solutions with the complementary basic solutions obtained in part a).

Problem # 2

Solve the following linear program by both the two-phases method and the big-M

Max \( 3x_1 - 3x_2 + x_3 \)

Subject to

\[ x_1 + 2x_2 - x_3 \geq 5 \]

\[ -3x_1 + x_3 + x_4 \leq 4 \]

\[ x_1, x_2, x_3, x_4 \geq 0 \]
Problem # 3

Consider the following problem

Min \(-101x_1 + 87x_2 + 23x_3\)

Subject to \(6x_1 - 13x_2 + 3x_3 \leq 11\)
\[6x_1 + 11x_2 + 2x_3 \leq 45\]
\[x_1 + 5x_2 + x_3 \leq 12\]
\[x_1, x_2, x_3 \geq 0\]

with optimal tableau

<table>
<thead>
<tr>
<th></th>
<th>(z)</th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(x_3)</th>
<th>(x_4)</th>
<th>(x_5)</th>
<th>(x_6)</th>
<th>(RS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(z)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-12</td>
<td>-4</td>
<td>-5</td>
<td>372</td>
</tr>
<tr>
<td>(x_3)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-2</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>(x_2)</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-1</td>
<td>9</td>
<td>-30</td>
<td>1</td>
</tr>
<tr>
<td>(x_1)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>19</td>
<td>-434</td>
<td>144</td>
<td>2</td>
</tr>
</tbody>
</table>

What is the solution of this linear problem obtained by decreasing the RS of the second constraint by 15?

By how much can the Right Side of the 2nd constraint increase and decrease without changing the optimal basis?

What is the solution of this linear program obtained by increasing the coefficient of \(x_1\) in the objective function by 25?

By how much can the objective coefficient of \(x_1\) increase or decrease without changing the optimal basis.

Would the current basis remain optimal if a new variable were added to the model with objective coefficient \(c_4 = 46\) and constraints coefficients \(A_4 = (12, -14, 15)\)?

Determine the solution of this linear program obtained by adding the constraint \(5x_1 + 7x_2 + 9x_3 \leq 50\).

Determine the solution of this linear program obtained by adding the constraint \(x_1 + x_2 + x_3 \geq 10\).

Determine the solution of this linear program obtained by adding the constraint \(x_1 + x_2 + x_3 = 30\).
Readings and other resources

Readings and other features of this unit are in readings and other course features list. https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio

https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio

https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio

https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio

https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio

https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio

https://docs.google.com/document/d/17XzMpGzpxaRYoPT464yEzL7RsYK1NK8wZwPMo-xUsl/edit - heading=h.44sinio
Unit 3. Transport, Assignment Problems, and Network Flow

Introduction to unit

In this unit, we put an emphasis on the application of linear programming problems. The transportation problem involves the application of an optimal transport of goods while the assignment problem involves the application of assigning people some tasks. The terminologies and the basics principles of graph theory are also introduced in this unit.

Graphs represent relations in a discrete set and not empty vertices and form a very useful mathematical framework to model a large class of discrete models of linear programming such as transportation problems, the assignment problem, minimum spanning tree, shortest path, network flow.

Also the main algorithms for solving these problems are presented.

Unit Objectives

After completing the unit, students should be able to:

- model and solve a transportation problem;
- model and solve a problem of assignment;
- calculate the minimum spanning tree of a network
- determine the shortest path between two vertices
- solve network flow optimization problems;

Key Terms

digraph: representation of a binary relation on a set of vertices

Vertex: elements entering the relationship or as a source or as a destination

Edge: pairs of related elements by origin-destination

Inner degree: number of edges bound on the vertex

Outer degree: number of edges originating at the vertex

Loop: edge to and from a same vertex

Symmetrical edges: edges to and inter-exchanged destination

Digraph: Graph Symmetric
Grade: number of edges adjacent to the vertex

Planar: graph represented in the plan where edges do not intersect

Regions: plan subsets defined by edges

Adjacent vertices: vertices connected by edges

Adjacent edges: edges connected to the same vertex

Adjacent regions: separate regions for some edges

Parallel edges: edges with the same source and same destination

Simple graph: no ties and no parallel edges

Simple vertex: degree of vertex is 2

Isomorphic: same representation other than labels of vertices and edges

Complete: with edge between any two vertices

Subgraph: graph obtained from another, deleting vertices and edges

Delete vertex: operation used for subgraphs

Path: sequence of vertices and adjacent edges

Reachability: vertices can be connected by paths

Adjacency: property of “proximity” between edges and vertices regions

Length: the number of edges on the path

Euler: path to a specific characteristic

Cycle: path to even start vertex and end

Hamiltonian: path that visits all vertices

Related: graph with path between any two vertices

Court: decomposition of a graph in two subgraphs eliminating only edges

cut (cut-set): set of removed edges to make the cut

Tree: connected graph without cycles

Root: Tree vertex with higher hierarchy
**Leaf:** degree of vertex 1

**Binary Tree:** Tree with two children per corner than leaf

**Spanning tree:** tree that connects all vertices

**Bipartite:** edge only between distinct subsets of the partition of the vertices

**Euler Formula:** \( v - a + r = 2 \)

**Elementary division:** simple vertex introducing an edge

**Homomorphic:** isomorphic except for elementary subdivisions

**Kuratowski Theorem:** a graph is either planar or homeomorfo \( AK5 \) or \( K3,3 \)

**Dual graph:** graph with an edge by adjacent regions map

**Dual map:** with a region of the graph vertex

**Color:** assigning colors (labels) the different adjacent regions

**Chromatic number:** minimum number needed to colour a map

**4-color theorem:** The minimum number needed to colour a map is 4

**Networks:** weighted graphs or digraphs

**Weighting:** assigning a weight (value) every edge or every vertex

**Storage:** accumulation values vertices

**Flow:** transfer of value from one vertex to another

**Transportation problem:** runoff and bipartite graph minimizing the total weight

**Assignment problem:** runoff and bipartite graph with unique edges

**Balanced:** total vertices we offer equal to total attempts vertices us

**Top left corner:** algorithm initialization method MODI or Stepping Stone

Minimum cost: algorithm initialization method MODI or Stepping Stone

**Fee:** algorithm initialization method MODI or Stepping Stone
Stepping stone: optimization algorithm on the transportation problem

Vogel / MODI: optimization algorithm on the transportation problem

Allocation: assigning values to cells in the Stepping Stone / MODI or Hungarian algorithm

Basic position: allocated cells

No basic position: reset cells

Alternating cycle: alternating horizontal step with vertical step to go back to the source

Degenerate solution: basic position allocated with zero

Hidden variable: Variable basic position allocated with zero

Small epsilon: variable “near zero”

Artificial point of supply: an artificial vertex connected to the supply of vertices

Artificial point search: an artificial vertex to which it binds all of them looking for vertices

Hungarian Algorithm: resolution method of assignment problem

Learning activities

Activity 3.1 - Terminology and basic elements of the graph theory

Introduction

Through the basics and terminology of graph theory; Graphs represent relations in a discrete set of vertices and not empty and as such form a very useful mathematical structure, similar for example ace functions, which also are relations of a set (domain) to another (codomain).

A large class of discrete models of linear programming can be represented by graphs as can be seen in the next activities.

So it is very important to know the terminology, the main theorems (Euler, Kuratowski and 4 colors) and the main problems (Euler cycle and hamiltonian cycle).
Activity Details

A directed graph or digraph is a representation of a binary relation on a set of vertices.

Example:

Given a set of vertices \( V = \{a, b, c, d\} \), the \( G = \{(b, a), (a, b), (c, b), (b, b), (d, c), (b, a), (c, d)\} \)

can be represented by the digraph

![Digraph Diagram](image)

Each ordered pair relationship is referred to as the edge graph \( G \) and is usually identified with an alphanumeric symbol.

Formally, a digraph is then defined by \( G = \text{data structure} (V, A, g) \) characterized by two sets \( V, A \) and table or \( g \ function: A \rightarrow V \)

Vertex: \( V = \{a, b, c, d\} \)

\( A = \text{Edges:} \{1, 2, 3, 4, 5, 6\} \)

Table

<table>
<thead>
<tr>
<th>( V )</th>
<th>( A )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( (2a, 2) )</td>
<td>( b )</td>
</tr>
<tr>
<td>( (b, 3) )</td>
<td>( b )</td>
</tr>
<tr>
<td>( (b, d) )</td>
<td>( 4 )</td>
</tr>
<tr>
<td>( (c, d) )</td>
<td>( 5 )</td>
</tr>
<tr>
<td>( (c, b) )</td>
<td>( 6 )</td>
</tr>
<tr>
<td>( (b, c) )</td>
<td>( 7 )</td>
</tr>
</tbody>
</table>

The digraph the edges has a source vertex: first pair of collating element, and a target vertex: second pair of collating element.

The number of edges entering a vertex called the inner degree of a vertex and the number of edges leaving a vertex called the apex outer degree. When the source and destination overlap, the edge is called a loop.
In the above example, the edge 2 is a bond.

Two edges are symmetric to the origin and fate of one is the destination and origin of each other, respectively.

In the above example, the edge 5 of the edge 6 is symmetrical.

When all the edges have symmetrical edges the digraph is said to be symmetrical and is called only graph.

Then the graph is a symmetric digraph.

[Diagram of a graph with labeled vertices and edges, including the symbols for vertices a, b, c, d, and edges 1, 2, 3, 4, 5, 6, 7, showing the symmetrical relationships and degrees of vertices.]

The degree of a vertex is the number of edges that connect to the vertex, and \( \text{degree} = \text{inner degree} + \text{outer degree} \).

In the example above, the degree of (b) 6.

A geometric representation of a graph makes matching a point of a geometric space with a vertex in the graph and a solid line between two geometric points with an edge between these points.

The two examples above are geometric representations of a digraph and a graph in the role of the plan (this sheet).

A graph is planar if there is a geometric representation of the graph in a plane (two dimensions) such that the edges do not intersect.

A planar graph separates the plane into distinct regions (indoor and outdoor).

For example, \( K_4 \) define 4 regions in the plane, three inner and one outer:

[Diagram showing a planar graph with 4 regions labeled Region 1, Region 2, Region 3, and Region 4, illustrating the separation of the plane.]

Two vertices are adjacent if they are extreme edge of the same, for example, are adjacent vertices a and c are as extreme edge 7:

[Diagram showing two adjacent vertices a and c connected by edge 7.]

Two edges are adjacent if they bind to the same vertex, for example, edges
1 and 7 are adjacent since bind to the same vertex:

Two regions are adjacent are separated by an edge, for example, regions I and II are adjacent they are separated by the edge 5:

Two edges are parallel if they have the same extremes.

Simple graphs have no loops or parallel edges.

Two graphs are isomorphic if they have identical representations unless lettering.

Handle without breaking the edges of their adjacent vertices resulting in isomorphic graphs, for example:

A graph is complete if all distinct vertices are adjacent.

Example of complete graphs with $n$ vertices, called $K_n$
A structure $H = (V', A', g')$ is the subgraph of graph $G = (V, A, g)$ if

$V' \subseteq V$

$A' \subseteq A$

$g' = g \mid _{V'}$ in a constraint $g$ to the subset $A'$

One important way to build subgraphs is deleting vertices and edges that are adjacent.

For example, if one of vertices $K_5$ is eliminated along with all the edges that connect to it, follows a graph $K_{5-1}$ (subgraph $K_5$).

A path from a vertex $n_0$ to a vertex $n_k$ is a finite sequence of vertices and edges, starting at $n_0$ and ending in $n_k$.

The vertex $n_k$ is said to be reachable from the vertex $n_0$ if there is a path from $n_0$ to $n_k$.

A graph can be represented by an adjacent matrix, with rows and columns and represented by the vertices input 1 if the vertices are adjacent and 0 if not:

The length of a path is the number of edges in this way.

In the example defines the path $(a, 7, c, 5, b, 3, d)$ red, with the length equal to 3:
An Eulerian path in a connected graph G is a path that visits each graph edge only once, (c, 7, a, 1 b, 2 b, 5 c, 6, b, 3 d, 4, c):

Euler was the first to note that if a graph has only one or more than two then odd degree vertices can not be a Eulerian path, otherwise there is always (to investigate the problem of Konigsberg Bridge).

The problem of Euler path is also known as the Chinese postman problem: what is the shortest way to go through all the streets without repeating?

A cycle is a path from one vertex \( n \) to the same vertex \( n \)

For example (c, 5 b, 3 d, 4 c) is a cycle of length 3

A Hamiltonian cycle in a graph G is a connected a cycle that visits each vertex of the graph exactly once (c, 7, a, 1, b, 3 d, 4, c):

Finding Hamiltonian cycles in an arbitrary graph is an NP-complete problem, then we can not know if there is a Hamiltonian cycle (or not) before the find (or not) by extensive search.

The problem Hamiltonian cycle is also known as the traveling salesman problem: what is best route to reach all households without repeating paths, returning to the starting house at the end of the day?

A graph is connected if there is a path between any two vertices.

A cut is a partition of the vertices of a graph into two subsets (disjoint whose union is the set of all vertices)

A cut generates two subgraphs disconnected, and a cut (cut-set) which is the set of all edges in a vertex with another vertex subset and the second subset of the partition.

The edges of the cut cross cutting.
A tree is an acyclic, connected and finite graph.

It is customary to depict with the root at the top and bottom sheets:

A tree of \( n \) vertices (nodes) has \( n-1 \) edges, and often defines a tree as a connected graph with \( n \) vertices and \( n-1 \) edges.

The depth of a vertex is the path length from the root to the vertex in question.

The height of a tree is the largest depth of all their vertices.

The most famous tree is the binary tree where each vertex parent has two children-vertices.

A spanning tree of a graph is a tree containing all vertices of the graph.

A minimum spanning tree of a weighted graph is a spanning tree with the lowest total weight, for example if the number of edges represent the weighting then the minimum spanning tree is given by:

A bipartite graph is the set of vertices can be partitioned into two subsets and the vertices of a same subset are non-adjacent.

A bipartite graph is complete if any vertex of one of the subsets is adjacent to all vertices of the other subset.

A well known full bipartite graph is \( K_{3,3} \)

This graph \( K_{3,3} \) is often used as an example of the inability to connect electricity, sewage and water and 3 adjacent houses, without crossing any of the lines:

<table>
<thead>
<tr>
<th>Electricity</th>
<th>Sewage</th>
<th>Water</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>How to connect electricity and water to the middle house</td>
<td></td>
</tr>
</tbody>
</table>

Euler's formula:
For a finite graph, planar and related worth the following identity (Euler):

\[ v \cdot a + r = 2 \]

where

- \( v \) is number of vertices
- \( r \) is the number of regions
- \( a \) is the number of edges

As an example, in the graph:

\[ a = 7, v = 4 \text{ and } r = 5 \text{ and Euler's formula holds: } 5 + 2 = \]

A vertex is simple a 1 has grade 4-7.

A vertex is a leaf has grade 1.

A vertex is simple if you have grade two (a digraph if inner degree = outer degree = 1).

An elementary subdivision of an edge is the introduction of a single vertex in the body of this edge.

Two graphs are homeomorphic if both can be obtained in the same graph as a sequence of elementary subdivisions.

Theorem of Kuratowski (characterization of non-planar graphs):

A graph is non-planar if and only if it contains a subgraph homeomorphic to \( K_{3,3} \) or \( K_5 \).

A planar map is a subdivision plan in enclosed areas.

The dual graph of a planar map is obtained by placing a vertex in each region (closed) of the map and an edge between vertices if their regions are adjacent.
[The text from the image is too large to be transcribed accurately.]

Reciprocally, because a graph can be constructed corresponding dual map.

The coloration of a map is the assignment of a symbol (a color) to each vertex of the dual graph without adjacent vertices remain in the same color.

Chromatic number of a graph is the smallest number of symbols needed to obtain a map of the dual staining.

Theorem of four colors:

The chromatic number of a simple graph, connected and planar (scp) is at most 4.

This theorem confirms that you can use only 4 colors to color any map plan, however great the number and complexity of the regions. Networks are weighted graphs or digraphs where each edge and/or vertex has a weight assigned by a weighting function:

\[ p: \mathcal{A} \rightarrow \mathbb{R} \quad \text{and} \quad A: p: \mathcal{V} \rightarrow \mathbb{R} \]

The weighting of the vertices on a network is referred to as storage capacity of vertex \( v \) and the weighting of the edges is referred to as flow capacity of the \( a_{ij} \) edge:

Graphs networks are widely used in modeling problems where one needs to analyze a relationship in a given set. Example:

**Conclusion**

After defining the terminology, concepts and the basic notation of graph theory, theorems of Euler and Kuratwoski and graphs were used for modeling the transport and assignment problems which will be resolved in the following unit.

**Activity 3.2 - Transport Problems**

**Introduction**

This activity is to analyze transport problems used as templates for runoff products from various points of origin (supply) to multiple destination points (search), and each origin to each destination a specific transport cost (penalty).

Transport problems can be solved by the Simplex or the SOLVER, but for problems with many decision variables may want to use alternative algorithms:

i) to find a feasible basic solution: top left, minimum cost or Vogel (penalty)

ii) to optimize the solution viable method of “stepping stone” or MODI
Various situations are analyzed, namely the facilitation of the solution when there are cycles, the existence of hidden variables and unbalanced problems.

**Activity Details**

The transportation problem is modeled by \( n \) the origin points \((i)\) and an amount \(a_i\) to be drained from each source, \(m\) target points \((j)\) and a quantity \(b_j\) to be supplied to each source at a cost of transportation from each source \((i)\) to each destination \((j)\) given by \(c_{ij}\).

The problem is said to be balanced when the total flow of all the centers of origin equals the total to provide all target centers:

\[
\sum_{i=1}^{n} a_i = \sum_{j=1}^{m} b_j
\]

The problem is to minimize the cost of transportation between the points of origin and destination points.

The transportation problem can be formulated as a linear programming problem:

Decision variable:

\( x_{ij} \) - amount to carry from the origin \((i)\) to the destination \((j)\).

Minimize the objective function:

\[
z = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij}
\]

subject to the constraints:

\[\sum_{j=1}^{m} x_{ij} \leq a_i \forall i\] - can not move from the origin \((i)\) more than \(a_i\)

\[\sum_{i=1}^{n} x_{ij} \geq b_j \forall j\] - \(j\) have to meet demand at the destination \((j)\) which is \(b_j\)

\(x_{ij} \geq 0\)

**Example**

Consider a production and sales of table water company which produces at three centers of production / origin \((a_1, a_2, a_3)\) and sell in four sales centers / destination \((b_1, b_2, b_3)\). The shipping cost from the origin \((i)\) to the destination \((j)\) is given by \(c_{ij}\).

The company is interested in knowing how many units should carry any of the centers of production for any of the sales centers to minimize transportation costs.

Data are the following:

From

\[(a_1, a_2, a_3) = (12, 7, 10)\]
Unit 3. Transport, Assignment Problems, and Network Flow

Destination:
\[(b_1, b_2, b_3, b_4) = (7, 8, 5, 9)\]

Shipping Cost, \(c_{ij}\):

<table>
<thead>
<tr>
<th>(c_{ij})</th>
<th>(b_1)</th>
<th>(b_2)</th>
<th>(b_3)</th>
<th>(b_4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_1)</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>(a_2)</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>(a_3)</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>

There are 12 decision variables \(x_{ij}\):

<table>
<thead>
<tr>
<th>(x_{ij})</th>
<th>(b_1)</th>
<th>(b_2)</th>
<th>(b_3)</th>
<th>(b_4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_1)</td>
<td>(x_{11})</td>
<td>(x_{12})</td>
<td>(x_{13})</td>
<td>(x_{14})</td>
</tr>
<tr>
<td>(a_2)</td>
<td>(x_{21})</td>
<td>(x_{22})</td>
<td>(x_{23})</td>
<td>(x_{24})</td>
</tr>
<tr>
<td>(a_3)</td>
<td>(x_{31})</td>
<td>(x_{32})</td>
<td>(x_{33})</td>
<td>(x_{34})</td>
</tr>
</tbody>
</table>

and it becomes time consuming, even for this little problem, use the Simplex table.

and the excel SOLVER: one obtains the optimal solution (full) given by:
\[
x_{11} = 0, x_{12} = 0, x_{13} = 5, x_{14} = 7
\]
\[
x_{21} = 0, x_{22} = 7, x_{23} = 0, x_{24} = 0
\]
\[
x_{31} = 7, x_{32} = 1, x_{33} = 0, x_{34} = 2
\]

\[
z = 5x_1 + 7x_3 + 7x_1 + 7x_2 + 1x_2 + 2x_3 = 55
\]

Because of the particularity of the above problem, you can solve it with more efficient alternatives, first finding a basic feasible solution and then iterating to find the optimal solution.

1. Stage I - find feasible basic solutions with one of the methods:
   - the upper left corner method
   - the minimum cost method
   - method of penalty (or Vogel)

2. Stage II - obtain the optimal solution one of the methods

método do stepping stone
**Método MODI**

The data of the previous problem can be organized in a table, having:

- in the upper left corner of each cell (red), transportation costs $c_{ij}$
- in the last column, the total demand for each sales center
- in the last line, the total supply for each production center

The cell in the lower right corner balancing: demand equals supply.

The aim of the announced method is a first step to make allocations \((x_{ij})\) to cells in order to obtain a basic feasible solution (satisfying the total of the last column and last row) and in a second step search for the optimal solution (lowest value of the objective function).

Allocated positions correspond to basic positions of the non-allocated positions and decision variables correspond to non-basic positions (values to zero).

**Method of the upper left corner**

One begins to fill the framework of cell allocation with the “upper left corner” in line a1 and b1 column.

The offer for a1 is 12 and b1 is looking for 7.

Then the cell a1 x b1 can receive maximum 7 just getting the offer of the a1 12-7 = 5 and the search for the b1 7-7 = 0. As demand column b1 has been exhausted, the remaining cells of the column are blocked (gray) and the total values of the row and column are adjusted accordingly:

And as the supply line a1 is not yet exhausted, you go to the column b2 where 5 remaining units of a1 will be used and the demand for b2 is reduced in 8-5=3. The remaining cell line a1, They are blocked:

In line b2 demand is satisfied on the line a2 and supply decreased by 7-3 = 4. The remaining cells are blocked in column b2:

The supply line a2 is not yet exhausted, you go to the column b3 where 4 remaining units of a2 will be used and the demand for b2 is reduced to 5-4=1. The remaining cell line of a2 are blocked:

On the line b3 demand is satisfied on the line a3 and decreased supply in 10-1=9

Finally, in the cell line a3 and column b4 supply and demand are equal:

The process results in a basic solution with 6 practicable allocations:

\[
z = 73 + 55 + 31 + 41 + 13 + 93 = 83
\]

but still we do not know whether this solution is good.
Method of minimum cost

In the method of minimal cost, instead of starting at the top left, begins the procedure with the cell that has the lowest cost. In the case of a tie the choice is arbitrary between the lower cost.

Choosing a cell line a1 and column b2 whose shipping cost is 1

and running the same supply reduction procedures and demand, we obtain:

The cell line a2 and b2 column with shipping cost 1:

The cell line a3 and b2 column with shipping cost 2:

The cell line a1 and column b4 with shipping cost 3:

The cell in the row a1 and column b4 with shipping cost 3:

and the end cell in the row a3 and column b4 with transportation cost 2:

The process results in a basic feasible solution with 6 allocations:

\[ z = 51 + 73 + 71 + 72 + 12 + 23 = 55 \]

but still do not know if it is the optimal solution.

This method looks at minimal cost position in each iteration and so the basic feasible solution found is better than the top left corner of the method that uses only pre-determined positions.

Method of penalties (or In Vogel)

In each row (and in each column) if you cannot allocate any demand or any offer at minimum cost position, and if there is another cell with the same cost on the same line (or the same column), there will be an increase the objective function, i.e., a penalty.

The method starts at a minimum cost position, e.g., \((a_1,b_3)\)

Calculates the maximum penalty in each row and each column and tries to avoid them:

In a1 line if demand or supply does not fit all in lower-cost cell \((a_1,b_3)\) then the next cell in that row with the lowest cost is \((a_1,b_1)\) or \((a_1,b_4)\), and the penalty cost is 3-1 = +2

In line a2 the lowest cost is 1 in two cells, then it does not fit all in a cell will fit in another at the same cost, then the penalty cost is 0, and continuing in this way are calculated the penalties for all rows and columns.

Identifies the row (or column) of major penalty and allocates the maximum possible value of the supply or demand in lower-cost cell \((a_1,b_3)\) this row (or column), the total set-up, block up the forbidden cells and recalculate again the penalties:

The biggest penalty is +1 and occurs in the 2nd row and the 2nd column. Choosing any, in this case the 2nd line, allocates the maximum amount of supply or demand the lower cost cell (a2, b2) of this row (or column), the total is settled, the cells were block-prohibited and recalculate to the new penalties:
The biggest penalty is +3 and occurs in the 2nd column. Allocates the maximum possible value of the supply or demand at the lowest cost cell (a3, b2) of this column, the total is settled, block up the forbidden cells and recalculate the new penalties:

The biggest penalty is +1 and occurs in the 1st column. Allocates the maximum possible value of the supply or demand at the lowest cost cell (a3, b1) of this column, the total is settled, block up the forbidden cells and recalculate the new penalties:

The penalties are all zero, you choose any of the lines or columns missing (has the same cost as the penalties are nil), allocates the maximum possible value of the supply or demand at the lowest cost cell (a1,b4) this line, the total set-up, block up the forbidden cells and recalculate again the penalties:

The process results in a basic feasible solution allocations 6:

\[ z = 51 + 73 + 71 + 72 + 12 + 23 = 55 \]

but still do not know if it is the optimal solution.

Because this method uses not only the demand at minimum cost position in each iteration but also the penalty, it is expected that the basic feasible solution found to be better than the method of minimal cost or the upper left corner.

**Alternating Series (Series)**

In relation to the above allocation, consider a graph with vertices in all allocated position (basic position), and where the edges correspond to all vertical or horizontal connections between the vertices. An alternating cycle (from now on only cycle) is a path that gets moving between two adjacent vertices, vertically or horizontally alternately, beginning and ending in the same cell. Let (n) be the number of points of origin and (m) the number of destination points.

Cycles may be even less than \( m + n - 1 \) frame allocations in cells, but with more than \( m + n - 1 \) allocations is inevitable that there at least one cycle. Case less than \( m + n - 1 \) cycles without allocations correspond to degenerate solutions with basic positions filled with 0 (hidden variable). The three methods of 1st stage ensure a basic feasible solution without cycles, i.e., no hidden variables.

The existence of cycles reduces the number of allocations.

**Example**

Consider the example of a viable solution with the following value of the objective function:

\[ z = 33 + 41 + 53 + 31 + 43 + 42 + 52 + 13 = 64 \]

8 allocations (variable base) and then, with several cases of different cycles:

1st case

or a 2nd case:

or a 3rd case:
To reduce the number of allocations, for example in the 3rd case, the lowest value (1) of the loop apex may be alternately added and subtracted to all the vertices of the cycle: and thus cancel the cell value (a₃, b₃), without affecting the total demand (columns) or supply (lines) and reducing the number of allocations from 8 to 7:

In this case it is still possible to reduce the number of allocations because the basic feasible solution that has still been 7 allocations cycle.

Subtracting and adding the lowest value of the cycle of vertices (2), we obtain:

which is a basic feasible solution with 3 + 4-1 = 6 allocations and with a value of different objective function:

$$z = 53 + 51 + 23 + 73 + 22 + 82 = 67$$

It concludes that the addition and subtraction of values to the vertices of a cycle changes the value of the objective function but does not change the overall demand and supply.

**The stepping stone method**

The stepping stone method makes use of the finding made in the last paragraph: the value of the objective function can be changed (optimized) without changing the overall demand and supply, introducing cycles and adding / subtracting values to the vertices of these cycles.

Example:

How to optimize the basic feasible solution of 3 + 4-1 = 6 allocations and z = 67 found in the previous example? If this is not the optimal solution, then some of the non-allocated positions must be allocated!

Starting with the first cell is not allocated, (a₁, b₂), the “stepping stone” algorithm calculates each position at which the unallocated variation of the objective function is entered if (+1) unit in position. To not affect the overall demand and supply will have to be added and removed in a drive cycle vertices created with this new entry:

To calculate the variance of the resulting objective function of adição / subtraction of a unit in the cycle of vertices created, simply add / subtract the costs in the cells of the respective vertices:

$$+5-2+2-3= +2$$

That is, the objective function increases (+2) with adding / minus one unit at the vertices of the created cycle.

Calculating the change in the objective function in the other non-allocated cells is obtained:

Is chosen cell (a₂, b₂) with minor variation (-2) and seeks the maximum value (θ) which can be introduced in that cell without obtaining negative inputs analyzing the cycle that gave rise to this variation:

The maximum amount of θ that does not result in negative entries is determined by the cell $$(a_{1}, b_{1})$$ given by $\theta = 5$.  

101
Replacing \( \theta = 5 \) for all vertices of the indicated cycle results in a new basic feasible solution.

The value of the objective function suffers a variation of \( z = (-2) \cdot 5 = -10 \), which is caused by the variation increment of one unit, multiplied by 0 units.

Calculating the new value of the objective function, we obtain:

\[
z = 51 + 73 + 51 + 23 + 72 + 32 = 57
\]

As still not known whether this solution is good, the process is repeated.

Calculating the variation of the objective function for adding / removing a unit on all non-allocated cells is obtained:

Is chosen cell \((a3, b4)\) with less variation (-1) and analyzes the cycle that gave rise to this variation:

The maximum value that does not lead to negative inputs is determined by the cell \((a2, b4)\), given by \( \theta = 2 \).

Replacing \( \theta = 2 \) all vertices of the indicated cycle results in a new basic feasible solution:

The value of the objective function suffers a variation of \( z = (-1) \cdot 2 = -2 \), which is the variation caused by the increase of one unit, multiplied by \( \theta \) units.

Calculating the new value, we obtain:

\[
z = 51 + 73 + 71 + 72 + 12 + 23 = 55
\]

As yet it is unclear whether this solution is great, repeat the process again.

Calculating the variation of the objective function for adding / removing a unit on all non-allocated cells is obtained:

It is noted that any change in the non-allocated cells results in an increase in the objective function, then the optimum value has been reached and the process ends.

The optimal solution is given by:

\[
\begin{align*}
x_{11} &= 0, \quad x_{12} = 0, \quad x_{13} = 5, \quad x_{14} = 7 \\
x_{21} &= 0, \quad x_{22} = 7, \quad x_{23} = 0, \quad x_{24} = 0 \\
x_{31} &= 7, \quad x_{32} = 1, \quad x_{33} = 0, \quad x_{34} = 2 \\
z &= 51 + 73 + 71 + 72 + 12 + 23 = 55
\end{align*}
\]

**Hidden Variable**

If the basic feasible solution is degenerate then there are less than \( m + n-1 \) allocations and hidden variables.

To overcome this problem, does an allocation of a small value “small epsilon \( \epsilon \)'' to a cell non-basic and proceeds with the iterating until basic feasible solution ceases to be degenerate, if possible.
the “small epsilon” (\( \varepsilon \)) is a value added to or subtracted from any real number different from zero, this number does not change: \( a \pm \varepsilon = a \)

Example

As an example, consider a transportation problem with the following data:

Origin:

\[
(a_1, a_2, a_3) = (40, 60, 50)
\]

Destination:

\[
(b_1, b_2, b_3, b_4) = (20, 30, 50, 50)
\]

Transport cost:

<table>
<thead>
<tr>
<th></th>
<th>( b_1 )</th>
<th>( b_2 )</th>
<th>( b_3 )</th>
<th>( b_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_1 )</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>6</td>
<td>8</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>( a_3 )</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

Using the upper left corner of the method is a basic feasible solution:

However, this basic feasible solution is degenerate as it has only five allocations instead of \( n + m - 1 = 3 + 4 - 1 = 6 \).

Applying stepping stone it appears that there cycle to increase the cell \((a_1, b_3)\) with \( \Delta z = +2 \), but not to the cell \((a_1, b_4)\):

Using “small epsilon” \( (\varepsilon) \) in a position (?) and a closed cycle, it becomes possible to calculate the variation \( \Delta z = +8 - 6 + 7 - 6 = +3 \):

Usually up continues until the cell \((a_2, b_4)\) where we have to use the “small epsilon” \( (\varepsilon) \) again:

Continuing in this way is the framework of the “stepping stone”:

Is chosen cell \((a_3, b_2)\) with minor variation (\(-1\)) and analyzes the cycle that gave rise to this variation: The maximum value that does not lead to negative inputs is determined by the cell \((a_3, b_3)\) is given by \( \theta = \varepsilon \).

Replacing all vertices of the indicated cycle results in a new basic feasible solution:

Note that the “small epsilon” simply passed from cell \((a_3, b_2)\) to a cell \((a_3, b_2)\).

In the new position, “small epsilon” enables the direct calculation of the variation for every unit increase in non-allocated positions and the consequent determination of the framework of the “stepping stone”: 

103
All variations are positive and the “small epsilon” remains on the table!

Means “one” optimal value of the objective function is obtained with only 5 allocations, a degenerate position:

\[
\begin{align*}
    x_{11} &= 20, x_{12} = 20, x_{13} = 0, x_{14} = 0 \\
    x_{21} &= 0, x_{22} = 10, x_{23} = 50, x_{24} = 0 \\
    x_{31} &= 0, x_{32} = \varepsilon = 0, x_{33} = 0, x_{34} = 50 \\
    z &= 20 \times 4 + 20 \times 6 + 10 \times 8 + 50 \times 6 + 50 \times 6 = 880
\end{align*}
\]

To confirm the results, make use of the SOLVER “Google sheets”:

where we find an optimal solution in different positions but with the same value of the objective function (!):

\[
\begin{align*}
    x_{11} &= 10, x_{12} = 30, x_{13} = 0, x_{14} = 0 \\
    x_{21} &= 10, x_{22} = 0, x_{23} = 50, x_{24} = 0 \\
    x_{31} &= 0, x_{32} = \varepsilon = 0, x_{33} = 0, x_{34} = 50 \\
    z &= 10 \times 4 + 30 \times 6 + 10 \times 6 + 50 \times 6 + 50 \times 6 = 880
\end{align*}
\]

This case indicates the great alternative existence, that is, two different sets of decision variables with the same great value of the objective function.

The existence of three zeros in the ‘stepping stone “indicates this fact: if allocation is made to any of the cells containing zero, the objective function does not change!

For example, if we make an allocation to the cell \((a_{23}, b_{31})\), and we proceed to another iteration and replacing the value of the vertices \(\theta = 1\) :

We find the exact value given by the SOLVER.

MODI method (modified distribution)

To calculate the positions with the greatest impact on the objective function, the MODI method uses the primal slack variables and dual \(s_{ij}, t_{ij}\) at where:

\[
i = 1, \ldots, m \text{ (n rows)}, j = 1, \ldots, n \text{ (n columns)}
\]

So also called ST method, UV method or LK method, as the name given to the primal and dual slack variables.

The MODI method follows the following procedure:

- calculates, for all the allocated positions, the values of \(s_{ij} + t_{ij}\) in such a way that \(s_{ij} + t_{ij} = c_{ij}\);
- calculates, for unallocated positions, the value of \(c_{ij} \cdot (s_{ij} + t_{ij})\);
- increments the cell with the lowest value \(c_{ij} \cdot (s_{ij} + t_{ij})\) and calculate the maximum value of \(\theta\) allowed for the values of the vertices of the resulting cycle.
adds that $\theta_{\text{max}}$ all vertices cycle for a reduction in the value of the objective function.

Example

Consider a transportation problem with the following data:

Origin: 

$$\begin{align*}
(a_1, a_2, a_3, a_4) &= (10,30,25,35) \\
(b_1, b_2, b_3, b_4, b_5) &= (10,15,20,25,30)
\end{align*}$$

Transport cost:

<table>
<thead>
<tr>
<th>$c_{ij}$</th>
<th>$b_1$</th>
<th>$b_2$</th>
<th>$b_3$</th>
<th>$b_4$</th>
<th>$b_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>7</td>
<td>2</td>
<td>9</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>$a_2$</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>$a_3$</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>$a_4$</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

Applying the upper left corner of the method we obtain a feasible basic solution with 7 allocations (no more than $4 + 5-1 = 8$ allocations) and without cycles:

where the header has been prepared for the calculation $s_{ij}$ and $t_{ij}$, initializing the position $s_{1} = 0$.

The allocated positions (non-zero basic), starting with the upper left corner calculate all values of $s_{ij}$ and $t_{ij}$ so as to satisfy the equation $s_{ij} + t_{ij} = c_{ij}$ for all $s$ value $i = 1, \ldots, m$ ($n^\text{th}$ lines), $j = 1, \ldots, n$ ($n^\text{th}$ columns).

For example in cell $(a_1 b_1)$ or easily-calculated value $t_{11}$

$$0 + t_{11} = 7 \implies t_{11} = 7$$

When there are $m+n-1=4+5-1=8$ allocations without cycles is achieved obtain all values of $s_{ij}$ and $t_{ij}$.

But in this case we have an allocation unless, that is a hidden variable, and there must be a "small epsilon" ($\epsilon$) to continue to calculate $s_{ij}$ and $t_{ij}$:

It can be calculated now using $s_{ij} + t_{ij} = c_{ij}$ (note that the values of the primal and dual slack variables may be negative) and all other already themselves $s_{ij}$ and $t_{ij}$ can be calculated:

In the non-allocated positions (non-basic), is calculated $c_{ij}(s_{ij} + t_{ij})$: 

105
Pick up the cell with the smallest negative value \( c_{ij} \), a cycle is created (using the "small epsilon" if necessary), and increases up to a value that does not render any negative value in cell cycle vertices:

Or maximum not resulting in negative inputs it is determined by the cell \( (a, b) \) and given by \( \theta_{max} = 10 \). Replacing \( \theta_{max} = 10 \) all the vertices of the indicated cycle, resulting in a new basic feasible solution (note that the "small epsilon" disappeared, may or may not return in subsequent iterations):

The next iteration, omitting steps, results:

Since all values are non-negative the procedure ends.

The optimal value of the objective function is:

\[
z = 10 \times 2 + 10 \times 6 + 5 \times 5 + 15 \times 5 + 5 \times 3 + 20 \times 4 + 5 \times 4 + 30 \times 2 = 355
\]

Achieved a great point:

\[
\begin{align*}
x_{11} &= 0, x_{12} = 10, x_{13} = 0, x_{14} = 0, x_{15} = 0 \\
x_{21} &= 10, x_{22} = 5, x_{23} = 15, x_{24} = 0, x_{25} = 0 \\
x_{31} &= 0, x_{32} = 0, x_{33} = 5, x_{34} = 0, x_{35} = 0 \\
x_{41} &= 0, x_{42} = 0, x_{43} = 0, x_{44} = 5, x_{45} = 30
\end{align*}
\]

It appears a 0 in cell (a2, b5) there are great alternative and so the look we want, we do another iteration increasing the cell:

The value of \( \theta_{max} = 15 \), that was substituted on the vertices cycle results in:

with the same great value of the objective function:

\[
z = 10 \times 2 + 10 \times 6 + 5 \times 5 + 10 \times 4 + 20 \times 35 \times 4 + 20 \times 4 + 15 \times 2 = 355
\]

but reached an alternative optimum:

\[
\begin{align*}
x_{11} &= 0, x_{12} = 10, x_{13} = 0, x_{14} = 0, x_{15} = 0 \\
x_{21} &= 10, x_{22} = 5, x_{23} = 0, x_{24} = 0, x_{25} = 15 \\
x_{31} &= 0, x_{32} = 0, x_{33} = 20, x_{34} = 5, x_{35} = 0 \\
x_{41} &= 0, x_{42} = 0, x_{43} = 0, x_{44} = 20, x_{45} = 15
\end{align*}
\]

Only for verification, using the SOLVER with this problem we obtain the same result:

Unbalanced problems

In cases where the total supply is different from the searching problem is said to be unbalanced. If the demand is higher than supply adds up an artificial point of demand which absorbs the difference.
If supply exceeds demand adds up an artificial point of offering that absorbs the difference.

The transportation costs of the edges that connect the artificial points to other points is zero, and therefore the transport of artificial points and does not affect the objective function.

With these artificial points of supply or demand, is solved the problem usually with the methods listed above.

In the final solution it is necessary to consider two cases:
the quantities intended for artificial point search are stored in its origins;
provenientes the amounts of the fictitious offer point is lacking in quantity received the destinations.

**Conclusion**

This activity was analyzed transport problems used as templates for runoff products from various points of origin (supply) to multiple destination points (search), and each origin to each destination a specific transport cost (penalty).

Transport problems can be solved by the Simplex or the SOLVER, but for problems with many decision variables may want to use alternative algorithms:

i) to find a feasible basic solution: top left, minimum cost or Vogel (penalty)

ii) to optimize the solution viable method of “stepping stone” or MODI

Several situations were analyzed, namely the facilitation of the solution when there are cycles, the existence of hidden variables and unbalanced problems.

**Activity 3.3 - Assignment problem**

**Introduction**

The assignment problem is a transportation problem with the additional constraint that every destination can only match a single source and vice versa.

This additional restriction is easy to set the problem of representation by graphs, so there are alternative algorithms, namely the Hungarian algorithm to solve assignment problems.

**Activity details**

The assignment problem is a transportation problem with the additional constraint that every destination can only match a single source and vice versa, hence the number of origins and destinations are equal.

A situation where this problem occurs is when assigning tasks or award of contracts, so also called problem of attribution or award.

Consider the problem already described the previous unit:
An agency Public Works launched a tender for the construction of $m = 3$ bridges $(a_1, a_2, a_3)$, and $n = 3$ contractor $(b_1, b_2, b_3)$ participated and all proposed a price $c_{ij}$, (penalty) for the construction of each bridge:

To minimize the risks, the Agency of Public Works intends to award a bridge to each company and would like to minimize the total amount of the award.

In this case, a basic feasible solution consists of

3 subgraphs $K_2$ disconnected, for example

It is a feasible solution $z = 896$

There are other feasible solutions, for example:

also is a basic feasible solution $z = 852$

What is the optimal solution?

Leave $m$ be the number of points of origin (supply) and $n$ be the number of destination points (demand).

The formulation of the assignment problem $m \times n$ requires min decision variables only indicate whether the edge between the source and destination exists:

$x_{ij} = 1$ if the bridge $i$ is awarded to contractor $j$

$x_{ij} = 0$ if the bridge $i$ is not awarded to contractor $j$

Minimize

$$z = \sum_{i=1}^{m} c_{ij} x_{ij}$$

constraints

$$\sum_{i=1}^{m} x_{ij} = 1$$

$$\sum_{j=1}^{n} x_{ij} = 1$$

$x_{ij} \in \{0,1\}$

It can solve the problem of the appointment using the Simplex or SOLVER.

Taking as example the previous problem and using SOLVER, we obtain:

with the optimum solution:

$$z = 120 \times 1 + 570 \times 1 + 160 \times 1 = 850$$

achieved the optimum:
However, the particular problem is more advantageous to use alternative algorithms, especially if there are a large number of decision variables.

In the above example, a simple observation motivates an alternative algorithm called Hungarian:

if all the contractors decide to increase the proposal of one of the bridges +1 unit, the decision of the Agency does not change;

if a contractor decides to increase the proposals of all bridges +1 unit, the decision of the Agency does not change;

Therefore, the initial problem can be reduced to an equivalent problem, by subtracting a fixed value to all cells of a column and subtracting another fixed value for all cells in a row.

After reduction, the simplified problem has at least one zero in each row and in each column, 1 and zero if there is a great value then this value is \( z = 0 \).

Soon the optimal solution, if any, will be in alternating positions (rows / columns) in which entries are 0;

Hungarian algorithm

Consider the problem stated above, with \( m = n = 3 \).

From the table of penalties:

Identifies the lowest value of the entries on each line \( (c_{\text{min}})_{i} \) and subtract this value to all entries of this line:

Is repeated now for the columns, identifies the lowest value of the entries in each column \( (c_{\text{min}})_{i} \) and subtract this value to all entries of this column.

The problem of low frame has at least 1 zero in each row and 1 zero in each column and has great value this value is \( z = 0 \).

The allocation of basic positions should be made first by lines and then by columns alternating cells whose value is 0:in the first line the cell \((a_{1}, b_{3})\) it is marked on the second line not if can mark the cell \((a_{2}, b_{3})\) because is in the same column than some already checked, logo mark-if the cell \((a_{1}, b_{3})\).in the third row can not select the cell \((a_{3}, b_{3})\) because is in the same column than some already checked, logo mark-if the cell \((a_{3}, b_{3})\).No need to repeat the process for the columns because they were already marked a cell columns / alternate rows.

The optimal solution \( z = \€ \) with the allocations \( x_{11} = 1, x_{12} = 1, x_{13} = 0 \) the base and the remaining variables are equal to .Applying the same reduced number of positions to the initial problem:
Obtain the optimum value of the objective function \( z = 120 + 570 + 160 = 850 \), which is equal to the value obtained using the SOLVER.

Often not immediately allocate the basic positions from the cells with 0, in which case must perform some additional procedures, best demonstrated with an example.

Example

Consider an invitation to tender to fill five jobs \((m = 5)\) that was answered by five candidates \((n = 5)\) where each candidate presents his proposed compensation in constant monetary unit the following table:

\[
\begin{array}{c}
(c_{max})_i \quad i \\
\end{array}
\]

Reduces the problem \((c_{max})_i\) i.e subtracting to each line inputs the lower line, and the same for the columns:

The basic positions are allocated:

- in the 1st line - the cell \(a_{1,1} \)
- in the 2nd line - the cell \(a_{2,2} \)
- in the 3rd line - you can not because it is a column with cell already allocated and there is no other 0 on the 3rd line
- in the 4th line - the cell \(a_{4,4} \)
- in the 5th line - the cell \(a_{5,5} \)

on the 3rd line - you can not because it is a column with cell already allocated and there is no other 0 on the 3rd line

- in the 4th line - the cell \(a_{4,4} \)
- in the 5th line - the cell \(a_{5,5} \)
- in the 1st column - already marked
- in the 2nd column - already marked
- in the 3rd column - already marked
- in the 4th column - already marked

the 5th column - is not possible because it is in line with cell already allocated and no other 0 the 5th column

It can not make all allocations:

How to overcome the problem?

From the obtained allocations can there be any other allocation to the following procedure:

Mark all unallocated lines

If any of the marked lines have 0, mark the columns where there are those 0’s

If the marked columns have any allocation, mark lines where there are these allocations

Repeat until no more appointments to do

Scratch with a dash above all unmarked lines and marked columns
Find the lowest amount of cells that have not been crossed out, designating it by 

Subtract the value of the cells not scratched 

Add $2 \times 6$ the value of crossed cells twice. 

11. Let as are the cells crossed only once The procedure results in a new framework, Make new allocations and repeat the procedure in the event of new lock. 

Using the previous example:

1) schedule lines without allocation 
2) select columns 0 on the marked lines 
3) mark the lines with allocation on the marked columns 
4) Repeat until there are no markings to do 
5) risk unmarked rows and columns marked 
6) Find the lowest value of the cells not scratched $= 10$ 

7,8,9) subtract the value of the cells not scratched add to the value of crossed cells twice, let as are the cells crossed only once. Make new allocations of basic positions in the new framework: 

in the 1st line - the cell $(a_1, b_1)$ 
in the 2nd line - the cell $(a_2, b_2)$ 
in the 3rd line - the cell $(a_3, b_3)$ 
in the 4th line - the cell $(a_4, b_4)$ 
in the 5th line - the cell $(a_5, b_4)$ 

All appointments are made: 

Applying the basic positions for the initial table: 

They obtain the optimal solution: 

$z = 90 + 90 + 120 + 70 + 100 = 470$ 

obtained in positions $x_{11} = 1, x_{23} = 1, x_{35} = 1, x_{42} = 1, x_{54} = 1$ 

Using SOLVER the same problem, we obtain: 

The same optimal solution.
Conclusion

The Assignment problem was presented, being a transportation problem with extra constraint, it can be solved with the methods already treated.

However with the additional restriction, it is easy to set the problem of representation by graphs, so there are alternative algorithms, namely the Hungarian algorithm to solve assignment problems.

This unit presented the procedures of this algorithm

Assignment

Answer the following questions with maximum clarity. It should be submitted after one week.

Unit Summary

The terminology, concepts and the basic notation of graph theory, theorems of Euler and Kuratwoski are presented. Graphs and the theorems were used for modeling the transport and assignment problems which will be resolved in the following unit.

This activity analyzes transport problems used as templates for runoff products from various points of origin (supply) to multiple destination points (search), and each origin to each destination for a specific transport cost (penalty).

Transport problems can be solved by the Simplex or the SOLVER, but for problems with many decision variables one may want to use alternative algorithms:

i) to find a feasible basic solution: top left, minimum cost or Vogel (penalty)

ii) to optimize the solution viable method of “stepping stone” or MODI

Several situations were analyzed, namely the facilitation of the solution when there are cycles, the existence of hidden variables and unbalanced problems.

Assignment problem was presented, that being a problem of transport with extra constraint that can be solved with the methods already treated. However the additional constraint is easy to set the problem of representation by graphs, so there are alternative algorithms, namely the Hungarian algorithm to solve assignment problems.

Unit Readings and Other Resources

The readings in this unit are to be found at course level readings and other resources.

This unit presented the procedures of this algorithm
Unit 4. Network Optimization

Unit Introduction

This unit deals with the network flow optimization problem and presents the basics and terminology of Markov stochastic processes.

features elements of Markov processes and the theory of queues.

presents applications and some case studies: Project Planning and game theory.

Unit Objectives

After completing this unit, you should be able to:

- determine the minimum tree in a network
- determine the path of least weight in a network
- determine the maximum flow in a network
- determine the minimum cost flow in a network
- model a waiting list problem, determine the average time and the average length of a queue.

Key terms

- **Kruskal**: algorithm for finding the minimum spanning tree
- **Prim**: algorithm for finding the minimum spanning tree
- **Dijkstra**: algorithm to find path of least weight between two vertices
- **Maximum flow**: maximum transportable through a network
- **Limiting capacity**: maximum transport capacity of each edge
- **Source**: vertex that is flooded
- **Sinkhole**: vertex that is drained
- **Transient**: flow time dependent state
- **Stationary**: State independent flow of time
- **Kirchhoff Balancing Act at the corners**: Total in = Total out
- **Direct flow**: flow toward the digraph (network)
- **Reverse flow**: flow amount can be decreased at the edge
- **Inverse edges**: edges with virtual reverse flow
- **Maximum flow**: Maximum amount that can traversal network
**minimum cut**: cut with less weight, separating source and sink

**Theorem Ford-Fulkerson**: optimality network: maximum flow is equal to the minimum cut

**minimum cost flow**: minimum weight flow in a network

fluent substance flowing

**overflow problem**: transportation passage of vertices

**vertex passage**: vertices with inner degree and a nonzero outer degree

**SIMPLEX network**: Simplex algorithm running on a network

**generalized transport**: generalization of transportation to any network

**Markov process without memory**: the current depends only on the previous immediately

**Queued**: accumulation zone at apex

**server**: service unit

**Kendall notation**: queues problem identification code

**random variable**: special function, the probability space for R

**Stochastic parameters**: parameters with values uncertain

**Transition probability**: probability of moving from one state to another

**Mesh step**: the distance between one state and the next

**Little equation**: relationship between probability and length of queue

**Activity on edge**: the planning model where the edge corresponding to an activity

**Activity on the vertex**: the planning model where the vertex corresponds to an activity

**Critical Path**: Path that determines the minimum plan of execution time

**NP-hard**: there is no resolution algorithm but there is verification algorithm results

**Early Finish (EF)**: off to end early
Learning Activities

Activity- Network Flows

Introduction

This activity presents trois types of network flow problems:

minimum spanning tree - Kruskal and Prim algorithms

Dijkstra algorithms

maximum flow - increased flow algorithms and the theorem of Ford-Fulkerson

This activity presents also the problem of minimum cost flow in a network and network analysis with multiple sources and sinks.
Activity Details

In the previous unit, transportation and assignment problems were solved with algorithms based on network modeling and then compared with the resolution of the same problems with the Simplex / SOLVER method.

Many other systems are modeled with graphs and networks.

In the following problems it is intended to analyze the flow problems in a network from one or more points of the origin (source) to one or more destination points (sink), each edge a given flow capacity and each vertex a certain capacity storage.

Minimum Spanning Tree

Let \( V \) be the set of villages in one country, with the distance between the villages mentioned in the weighted graph:

A telephone operator want to know what is the minimum length of telephone cables required to connect the villages with each other.

To solve this problem, there is a minimal tree, for cycles only create redundancy and increase the cost.

Kruska algorithm

It creates a list of all the edges

Sort to list the respective weights of the edges

It begins by the lower edge weight and in a process of elimination, one goes adding more edges to the tree, without creating cycles:
The total minimum weight tree generator of this example is 290 weight units, which is the minimum length of telephone cable which allows to connect all the towns.

**Prim Algorithm**

Alternatively, you can create a basic subgraph and extend it according to certain rules that result in the end to the desired minimum tree:

Create an A subgraph with an arbitrary vertex of the original graph;

The Include in a new edge and its vertex, according to the following rules:

be attached to any of the vertices already in A

less weight among all candidates;

will form cycles with those who are already in A

Repeat if you are not able to continue.

The resulting subgraph is the minimum desired tree.

Using the graph of the previous example and starting with the vertex r, we obtain:
The spanning minimum is the total weight of 290 (is equal to that obtained by the Kruskal method).

4.1.2 The Shortest Path

Example: A typical example of this problem is as follows:

Let be the number of airports in a country where an air carrier intends to study the costs associated with the different routes in order to minimize the cost of fuel shown in each weighted graph of the tour:

How to determine the shortest path (relative to weight) between the vertex-origin and the vertex-destination?

The method of “brute force” list all possible paths and determines the shortest route.

However, when the network increases the complexity of this method consumes too much computing resources and becomes inefficient. If the balance is non-negative, the Dijkstra’s algorithm provides satisfactory results:

1) Prepare a table of all the vertices in the header of the columns;
2)circulates the vertex-source and starts iterating through that corner in the 1st row;
3) Record below each column in the weights of the edges adjacent and non-adjacent brand was a symbol, ie \( \infty \)
4) The lowest weight recorded in line is circulated indicating the end of the iteration in this column and the vertex corresponding to it goes to the following line:

| a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | x |
| g | 18 | 10 | \( \infty \) | 17 | 18 | | | | | | | | | | | | | | | | | | | |

5) Repeat the procedure with the new vertex on the line, without forgetting to add the value circulating in the previous iteration for all weights of edges adjacent to the vertex in the new line:
The resultant value is less than the previously registered in the column, then replaces the previous register,

if the value is equal keep the former but brand yourself as an alternate path

if the value is higher keep the previous record

In circled records indicate that the respective column does not make any changes:

6) Repeat the process and if two equal values are found, one of them is chosen because the other will be revisited later,

7) The process continues until the target vertex be circulated:

The path found “arrived” a j coming from c because it was in the c line was the last change in the j column of 41; arrived a c come from e and because it was on the line of e and that was the last change in the c column from ∞ to 23; We came to e and coming from g as can be read directly on the column.
Thus, one can reconstruct the shortest path:

\[ g \rightarrow a \rightarrow b \rightarrow c \rightarrow j \]

with the length of 41 (weight units).

If there are markings for alternative paths, one can identify them “out” in the respective markings.

There may be other ways to go through j, but it must have length greater than 41.

The table shows the shortest paths from the vertex-origin g to the Circulars columns, as it was the case for j.

Continuing the frame to have all the columns get circulated to all shortest paths from the vertex-origin g:
For example, the shortest route, starting from g to get the x, runs:

\[ g \rightarrow r \rightarrow s \rightarrow t \rightarrow x \]

and has length 102:

**Maximum flow**

In the network flow problem rather than penalty (cost of ownership) \( c_{ij} \) by using the line \((i, j)\) as seen in transport and designation problems, have a limiting capacity \( u_{ij} \) flow associated with the edge \((i, j)\).

Example: A typical example of this problem is as follows:

Let \( V \) be the set of water tanks in various localities in the city, connected by a targeted network whose pumping capacity in each course is indicated in weighted digraph (m3/hour), with the arrows representing the direction of flow:
What is the maximum flow that can flow on the network (m³ / hour), from the source 1 to the sink 5?

The binary adjacent matrix, leaving without meeting the zero entries is given by:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The column 1 only zero entries indicate that this vertex is a source.

The line 5 only zero entries indicate that this vertex is a sink.

In the typical maximum flow problem, it is assumed that:

the transient has been exceeded and therefore do not consider local stores (at each vertex)
the steady state was reached: all the flow entering the source out the sink.

Quantity available at source is infinite;

Amount lost to the sink is infinite;

The objective is to maximize total z stream flowing on the network from the source 1 to the sink 5.

Each edge has a non negative capability \( u_{ij} \geq 0 \), where \( i,j = 1, \ldots, n \), and numbering of the vertices, and two vertices are not adjacent then \( c_{ij} = 0 \).

The flow at the edge is the decision variable \( 0 \leq x_{ij} \leq u_{ij} \), where \( i,j = 1, \ldots, m \) the number of vertices, and two vertices are not adjacent then the flow is \( x_{ij} = 0 \).

Because the system is stationary, Kirchhoff’s law applies: the total flow entering a vertex \( k \) is equal to the total flow leaving this vertex

\[
\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj}
\]
With this description, the maximum flow problem in a network is formulated by:

maximize \( z \) the flow entering the source and sink out in a steady state.

\[
\begin{align*}
   z &= \sum_{j=1}^{m} x_{ij} + \sum_{i=1}^{n} x_{in} \\
   0 &\leq x_{ij} \leq U_{ij} \quad \text{for} \quad i, j = 1, \ldots, l \\
   \sum_{i=1}^{l} x_{ik} &= \sum_{j=1}^{m} x_{jk} \quad \text{for} \quad i = 1, \ldots, m
\end{align*}
\]

subject to the constraint

This problem can be solved by the Simplex / SOLVER, but when the network is complex, there are alternative algorithms easier to perform.

Increased Flow Algorithm

The increased flow algorithm analyzes the ability of all possible paths of the source to the the sink, considering not only the direct flow of the edges but also the reverse flow of the edges if they already have some direct flow.

In this example, the maximum flow of each direct path is equal to the capacity limit of the line:

The path (1,2,4,5): 10 is maximum flow rate (mark up this flow below the capacity of each edge):

Flow=10

The path (1,2,3,5): Maximum flow 5 (mark up the additional flow in the edges that already had markings):

Flow=5

The path (1,2,3,4,5): Maximum flow 0 as the capacity of edge (2,3) is already depleted:

Flow=0

The path (1,2,4,3,5): includes reverse edge (4,3) but as there is no flow to reverse (for the direct flow is 0), the maximum flow in this way is 0:

Flow=0

The path (1,3,5): maximum flow of 5 (remaining capacity of (3,5)):

Flow=5

The path (1,3,4,5): Maximum flow 0:

Flow=0

The path (1,3,2,4,5): includes opposite edges (3,2) having an ability to reverse the flow of direct 5 existing, but as the ability of the edge (4,5) is already exhausted, the maximum flow rate of 0:

Flow=0

The path (1,3,2,4,3,5): reverse includes two edges (3,2) and (4,3) while the first and has an ability to reverse the flow of direct 5 existing, the second it has no direct flow then the
maximum flow rate of 0:

After analyzing all possible paths, the calculated flows are added to obtain the maximum flow of the network: $\max = 5 + 10 + 5 + 0 + 0 + 0 + 0 = 20$

2 - Minimum cut algorithm

The minimum cut algorithm analyzes the dual problem of increased flow and network analyzes all cuts that leave unconnected two subsets of vertices, one containing the source while the other contains the sink.

The total capacity of the cut or cutset (set of cut edges) is calculated taking into account the orientation of the forward and reverse edges:

edges: {\{1,2,\} (1,3): Cut Capacity 30 + 20 = 50

edges {\{1,3,2,3,2,4\}} Cut Capacity 20 + 5 + 20 = 45

Edges {\{2,4,3,4,3,5\}}: Cut Capacity 20 + 15 + 10 = 45

Edges {\{1,2,2,3,2,3,4,3,5\}}: Cut capacity 30 – 5 + 15 + 10 = 50 (note that the ability of the edge (2,3) has an orientation (-) representing a reverse Cut
edge: :Cut Capacity : (note that the ability of the edge has an orientation representing a reverse Cut

Of all the cuts, the minimum cut capacity is \( w_{\min} = 20 \).

The fact that the maximum flow of the network is equal to the minimum coincidence is not cut but the results from the primal-dual relationship and optimality conditions of a network is termed as the theorem Ford-Fulkerson.

To go a little deeper, this relationship determines the primal formulation of the problem given in the typical example

Maximize flow \( z \)

The expression \( z \) can be calculated using Kirchhoff's law at source: \( z = x_{12} + x_{13} \), or sink: \( z = x_{35} + x_{45} \)

However, there are advantages in treating \( z \) as another unrestricted decision variable in the signal.

Constraints given by the limiting capacity of the edges (equal to the number of edges) are:

\[
\begin{align*}
    x_{12} & \leq 30 \\
    x_{13} & \leq 20 \\
    x_{24} & \leq 20 \\
    x_{21} & \leq 5 \\
    x_{34} & \leq 15 \\
    x_{35} & \leq 10 \\
    x_{45} & \leq 10
\end{align*}
\]

Constraints given by Kirchhoff's law at each vertex (equal to number of vertices):

Vertex 1, \( \text{out} \ (+), \text{in} \ (-) \):
\[ x_{12} + x_{13}z = 0 \]

Vertex :

\[ -x_{12} - x_{23} + x_{24} = 0 \]

Vertex :

\[ -x_{13} - x_{23} + x_{34} + x_{35} = 0 \]

Vertex :

\[ -x_{24} - x_{34} + x_{45} = 0 \]

Vertex :

\[ -x_{15} - x_{45} + z = 0 \]
Restrictions given by the non-negativity of decision variables \( x_{ij} \geq 0 \) and \( z \) is left unrestricted in sign (although it is not always negative).

Summarizing the formalization of the primal:

\[
\max \quad z
\]

subject to the constraints

\[
\begin{align*}
-x_{12} + x_{13} + x_{15} &= 0 \\
-x_{12} + x_{21} + x_{24} &= 0 \\
-x_{13} + x_{23} + x_{34} + x_{35} &= 0 \\
-x_{24} + x_{34} + x_{45} &= 0 \\
-x_{35} + x_{45} + z &= 0 \\
x_{12} &\leq 30 \\
x_{13} &\leq 20 \\
x_{24} &\leq 20 \\
x_{23} &\leq 5 \\
x_{34} &\leq 15 \\
x_{35} &\leq 10 \\
x_{45} &\leq 10
\end{align*}
\]

Solving the primal using the SOLVER:
One obtains the same result as that obtained with the increased flow and a minimum cut.

To calculate the dual-problem (minimization of the cut) checks that the number of dual variables is equal to the number of constraints \((n + m + 1)\) and denote by \(i_j\) and \(h_{ij}\) where \(ij = 1, \ldots, n:\)

\[
\begin{align*}
& w_1: x_{12} + x_{13}z = 0 \\
& w_2: -x_{12} + x_{23} + x_{24} = 0 \\
& w_3: -x_{13} + x_{23} + x_{34} + x_{35} = 0 \\
& w_4: -x_{24}x_{34} + x_{45} = 0 \\
& w_5: -x_{35}x_{45} + z = 0 \\
& h_{12}: x_{12} \leq 30 \\
& h_{13}: x_{13} \leq 20 \\
& h_{24}: x_{24} \leq 20 \\
& h_{23}: x_{23} \leq 5 \\
& h_{34}: x_{34} \leq 15 \\
& h_{35}: x_{35} \leq 10 \\
& h_{45}: x_{45} \leq 10
\end{align*}
\]

Right Side (RS) as the objective function coefficients:

\[
\min w = \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ij}h_{ij} - w_{ij} + 30h_{12} + 20h_{13} + 5h_{23} + 15h_{34} + 10h_{35} + 10h_{45}
\]

collection’ vertical of coefficients results in a restriction for each primal variable

\[
x_{ij} - w_{ij} + h_{ij} \geq 0 \quad \text{for } ij = 1, \ldots, n \quad \text{with the preferred inequality minimizing (≥) as the variables of the primal are non-negative:}
\]

\[
\begin{align*}
& x_{12}: w_1 + h_{12} \geq 0 \\
& x_{13}: w_1 + h_{13} \geq 0 \\
& x_{24}: w_2 + h_{24} \geq 0 \\
& x_{23}: w_2 + h_{23} \geq 0 \\
& x_{34}: w_3 + h_{34} \geq 0 \\
& x_{35}: w_3 + h_{35} \geq 0 \\
& x_{45}: w_4 + h_{45} \geq 0
\end{align*}
\]

\[
z: w_1 + w_5 = 1 \quad \text{(as was considered unrestricted in sign)}
\]

\[
h_{ij} \geq 0 \quad \text{(because inequalities in their primal constraints are preferred maximizing)}
\]

128
(unrestricted in sign) because the restriction from the Kirchhoff’s law is an equality.

Solving the dual using the SOLVER:

we obtain the same result as that obtained with increased flow and minimum cut treated.

Minimum cost Network flow

All Optimization problems in network already treated are:

- transport
- designation
- shortest path
- maximum flow

There are special cases of the problem of minimum flow-cost network, which have not only a certain value of $c_{ij}$ use of the line or edge $(i,j)$ but can also have a $u_{ij}$ limiting capacity of each edge $(i,j)$ and vertices can be sources $b_{i} > 0$, passing by $b_{i} = 0$ or sink $b_{j} < 0$.

The fluent flowing on the network is measured in units (un), the flow $(x_{ij})$, the limiting capacity $(u_{ij})$, the input sources and output sinks in unit per time (un / time) the cost of using $(c_{ij})$ of the edges in monetary units per unit of fluent ($/un$).

In general, we set up a fixed time basis: hr, day, month or year to calculate the flow in order to manipulate the flow units.

In general, a network flow problem can have multiple source vertices, vertex multiple-sink and also various vertices passage.

The problem of minimizing the cost of the flow network is called a transshipment problem with some vertex of passage, satisfying all the constraints of the sources, sinks and capacity of the lines.

A typical overflow problem is as follows:

Consider a transshipment problem modeled by the following network (example taken from G.
Srinivasan, NPTEL):

where running costs in each edge are indicated in monetary units per container ($/a$).

The problem is no capacity limitation, $u_{ij} = \infty$, $a$.

Vertices (1,3) are sources with input $b_1=6$ and $b_3=4$ respectively, or vertex 2 which is passing $b_2=0$ and vertices (4,5) are sinks with negative input $b_4=-5$ and $b_5=-5$ monetary units per container.

What is the minimum flow network cost, satisfying all conditions?

Being a linear programming problem it can be solved using the SOLVER, the network SIMPLEX or a generalization of MODI method (or stepping stone) follows:

The primal problem is formalized as

Minimize $z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}x_{ij} = 8x_{12} + 6x_{13} + 5x_{24} + 7x_{25} + 6x_{24} + 3x_{35} + 4x_{45}$

Restrictions primal: (Kirchhoff’s laws with conservation, the vertices:

$\sum_{i=1}^{n} x_{ik} = b_k$

$x_{12} + x_{13} = 6$

$x_{12} + x_{24} + x_{15} = 0$

$x_{12} + x_{24} + x_{25} = 4$

$x_{24} x_{34} + x_{45} = -5$

$x_{35} x_{45} = -5$
Note that the sum of the left sides of constraints and the sum of the right sides of restrictions are both equal to zero, showing that the system is linearly dependent.

The non-negativity \( x_i \geq 0 \)

Applying SOLVER the primal is

\[
\begin{align*}
\text{Maximize} & \quad w = 6y_1 + 2y_2 + 4y_3 + 5y_4 + 5y_5 \\
\text{Dual constraints:} & \quad (y_i \leq c_{ij}) \\
& \quad y_1 + y_2 \leq 8 \\
& \quad y_1 + y_2 \leq 6 \\
& \quad y_2 + y_4 \leq 5 \\
& \quad y_2 + y_5 \leq 7 \\
& \quad y_3 + y_4 \leq 6 \\
& \quad y_3 + y_5 \leq 3 \\
\end{align*}
\]

variables \( y_i \) unrestricted in sign

Applying the SOLVER, the dual we obtain:

\[
\begin{array}{cccccc}
B & A & D & C & E & F \\
2 & 1 & 0 & 3 & 4 & 5 \\
1 & 0 & 4 & 1 & 5 & 3 \\
3 & 0 & 5 & 2 & 6 & 4 \\
4 & 0 & 6 & 3 & 1 & 5 \\
5 & 0 & 7 & 4 & 2 & 6 \\
\end{array}
\]
the optimal solution \( w = 81 \) obtained in \( y_1 = 9, y_2 = 1, y_3 = 3, y_4 = -3 \).

The problem of a minimum cost network flow can also be resolved using a Simplex algorithm in the network.

Network SIMPLEX

Starts with an arbitrary basic feasible solution to the primal problem:

\[
\begin{align*}
x_{12} &= 6, x_{24} = 6, x_{25} = 4, x_{45} = 5\\
z &= 8 \times 6 + 5 \times 6 + 6 \times 4 + 4 \times 5 = 122
\end{align*}
\]

Since there are \( n = 5 \) vertices, there are at most \( n-1 = 4 \) edges in the solution (there may be less if the solution is degenerate).

At the break complementarity theorem, in optimum condition we have: \( y_i^r y_j = c_{ij} \)

As the primal problem is linearly dependent of the dual variables is arbitrary and may be set to zero, and then fixed \( y_5 = 0 \) and others are calculated by the formula \( y_i^r y_i = c_{ij} \) for all basic edges.

For non-basic edges, checks if the condition \( y_i^r y_j c_{ij} \leq 0 \) or equivalently \( y_i^r y_j c_{ij} \leq 0 \) is violated:
The dual solution $y_1 = 17$, $y_2 = 9$, $y_3 = 10$, $y_4 = 4$, $y_5 = 0$ is not feasible because $y_j \leq c_{ij}$ condition is violated, then the primal solution is not optimal.

The edge with maximum value of $y_j$ enters the base with a stream, and flows in, the remaining edges have to be adjusted to find the highest value compatible with the law of Kirchhoff:

$$\theta = 4$$

It is the value $\theta = 4$ that sets up the flow for the second iteration.

It uses the new basic solution to repeat the procedure.

If there are less than $n-1$ edges in the basic solution then this is degenerate and introducing an edge base with “small epsilon” and then normally resolve.
In this example, after a few iterations, this is the optimal solution:

which is the same solution found by SOLVER

The Generalized Transport

Network Simplex method is an application of the method used in the transport problem: we use the northwest corner, minimum cost or Vogel to find a basic solution and then we use the stepping method stone or MODI to make iterations and optimize the solution.

Using the method of transportation problem in the previous example, it creates a framework similar to the transportation problem:

Entries \((i, j)\) represents the cost \(c_{ij}\) of the respective edge

if there is no edge to mark up entry “big M”, a large number compared to the numbers of problem.
in the line or marginal column total, mark up the input source or sink, both with positive values.

The problem is balanced with total extracted from the source \(6 + 4 = 10\) is equal to the total drained from the sink \(5 + 5 = 10\). It adds this value (10) to all cells of the marginal row and column:
making “artificially” all vertices are source and sink of a basic value of (10) in addition to their own values of the problem.

Now you can use the method of transportation for the next frame that lacks interpretation:

As the graph has no loops, the cost diagonally 0 and flows of diagonal edges do not influence the objective function. As an artificial value to every vertex 10 has been added, it is expected that the flow of diagonal edges is 10, except for passage vertices, in this case the apex end 3 where this value is lower than 10-6 = 4, showing which is less costly use this vertex in the transport 6 units to the other vertices.

The solution is given immediately by the non-diagonal values, for the optimal solution of $z = 81$.

**Conclusion**

In this unit, were presented four types of network flow problems:

- minimum spanning tree - Kruskal’s algorithm and Prim
- shortest route - Dijkstra algorithms
- maximum flow- increased flow algorithms and Ford-Fulkerson theorem
- minimum cost flow in a network and network analysis with multiple inputs and outputs.

The resolution algorithms were also presented.
Assignment

Exercice 1:
Consider the undirected network as shown in the figure below, find the set of edges connecting all nodes such that the sum of the edge length is minimized.

Exercise 2:
Using the same network, find the set of edges connecting all nodes such that the sum of the edge length is minimized.

Exercise 3:
Using the same network, find the set of edges connecting all nodes such that the sum of the edge lengths from the root to each node is minimized.

Activity Case study

Introduction
It presents case studies of applications of research methods in the previous units in planning projects and in game theory.

Activity details

Planning and project management
How to coordinate the execution of a project with multiple interdependent activities?
The procedure begins with the breakdown of project activities, identifying for each activity duration, (s) predecessor (s) immediately (s) and the events, which are the beginning and the end of any activity:

- activity
- duration
The precedence relationship allows us to develop a network where the vertices are events (beginning or end of an activity), the edges are the activities and the weight associated with each edge \((i, j)\) is the duration (time cost) \(d_{ij}\) of its activity.
The main difference between this problem with the network flow problem is that the duration \( d_{ij} \) of edge \((i, j)\) is a minimum and no maximum limit on the flow edge.

An activity has a certain minimum length, but it can take longer if there are delays.

What is the shortest possible time to carry out the project?

To resolve this issue we need to find the longest path between the start and end of the project, called the critical path.

Finding the critical path in a network is an LP-hard problem, and so we do not have a general algorithm for the solution but have an algorithm available to check if any path found is critical.

But in cases where the network is simple, as in the previous example, it is possible to use the critical path method (CPM), a direct passage:

Mark the starting vertex with value 0 (the red box)

marking the following vertex with the largest value of the previous marked vertices plus the time cost of edges ending vertex that:

the apex 4 arrive two edges, via D \(15 + 10 = 25\) and via E \(20 + 15 = 35\), the highest value is chosen because it is what determines the period of the whole project:
and proceeding in this way you arrive at the last vertex representing the completion of the project and the critical path is 70 time units, and critical activities are Baker Marked in red:

For activities that are on the critical path, any delay will affect the time of project implementation.

Note that this algorithm is similar to Dijkstra’s algorithm with the difference that instead of choosing the shortest route we choose the long way.

We can also run the CPM algorithm with a reverse pass, beginning at the last corner with 70 and subtracting the temporal cost of the return trip to reach the first corner (values in the green box):

It appears that all vertices that are on the critical path has the same value both in direct passage or the reverse passage $1 \rightarrow 3 \rightarrow 4 \rightarrow 6 \rightarrow 7$.

Moreover, some vertices in the two values do not match what it needs some explanation.
The value obtained in direct passage (red box) represents the earliest start values for i and j in order activity \( (i, j) \):

The value obtained with the reverse passage (green box) represents the later onset values of i and j in order activity \( (i, j) \):

For example, C activity between the vertices \( i = 2 \) and \( j = 5 \), with the indicated values:

The interpretation is as follows:

the early start of C activity may only be 15 units of time after the start of the project;
the Late Start from C activity may only be 55 units of time after the start of the project;
The Early Finish of C activity may only be 25 units of time after the start of the project;
The End C of the later activity can be 60 time units after the start of the project;
It defines the total free floating fluctuation and an activity \( (i, j) \) as:

For example, the C activity between vertices \( i=2 \) and \( j=5 \) is shown in the figure:

Floating total of C activity is 60-15-25=20
the free float of C activity is 55-15-25=15
The free float is always less than or equal to the total float.

The free float is free pad that the activity has to be delayed without affecting neither the full cost of the critical path nor even the critical path.

A top pad to the free float but less to the total float, also does not affect the total cost but the critical path can already go through other edges.

If the pad is greater than the total buoyancy then the activity \( (i, j) \) shall belong to the critical path to the total cost of changes accordingly.
The CPM problem is a linear programming problem and has a primal and dual formulation.

Primal formulation:

Decision variable

\[ x_{ij} = \begin{cases} 1 & \text{edge} (i, j) \text{ belongs to the longest path} \\ 0 & \text{on the contrary} \end{cases} \]

Maximize

\[ \sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij} x_{ij} \]

Restrictions (Kirchhoff's law at the vertices: If the critical path passes through the vertex so there is an inlet and an outlet which cancel except in the beginning that the total is +1 order and the -1 is full and if not passes through the vertex where the sum is also zero)

\[ x_{12} + x_{13} = 1 \]
\[ -x_{12} + x_{24} + x_{28} = 0 \]
\[ -x_{13} + x_{34} + x_{36} = 0 \]
\[ -x_{24} + x_{34} + x_{45} + x_{47} + x_{46} = 0 \]
\[ -x_{24} + x_{46} + x_{57} = 0 \]
\[ -x_{34} + x_{46} + x_{67} = 0 \]
\[ -x_{47} - x_{57} - x_{67} = -1 \]

As the restrictions form a unimodular system can be made \( x_{ij} \geq 0 \) and solve the primal problem with the Simplex or the SOLVER:

They obtain the same duration for the project and the same critical path.

The dual problem we obtain using the conversion rules:

Minimize \( w = y_{1} - y_{2} \)
Restrictions:

\[
y_1 - y_2 \geq 15 \quad y_5 - y_7 \geq 10
\]
\[
y_1 - y_3 \geq 20 \quad y_6 - y_7 \geq 20
\]
\[
y_2 - y_4 \geq 10
\]
\[
y_2 - y_5 \geq 25
\]
\[
y_3 - y_4 \geq 15
\]
\[
y_3 - y_6 \geq 20
\]
\[
y_4 - y_5 \geq 20
\]
\[
y_4 - y_6 \geq 15
\]
\[
y_4 - y_7 \geq 30
\]

They are unrestricted in sign.

The problem can be solved with the Simplex or the SOLVER.

Sometimes it is necessary to analyze problems in that the duration is a random variable and not a deterministic variable.

AON Exemple

The modern trend of planning and project management is doing the analysis with the activities at the corners.

As an example of a network of activities of a project in which the activity is placed at the apex, we use the example described in Chapter 9 of Lieberman, Hillier.

The company STROIKA has a maximum of 47 weeks to build a factory.

A penalty of 300 will apply for a period of default.

A bonus of 150 will be awarded for meeting the deadline.

Construction of the plant involves 14 activities whose duration and precedence is given in the table below:
### Activity Table

<table>
<thead>
<tr>
<th>Activity</th>
<th>Description</th>
<th>Immediate Predecessors</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Excavate</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>B</td>
<td>Lay the foundations</td>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>Put off the roof wall</td>
<td>B</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>Put off the roof</td>
<td>C</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>Install the exterior plumbing</td>
<td>C</td>
<td>4</td>
</tr>
<tr>
<td>F</td>
<td>Install the interior plumbing</td>
<td>E</td>
<td>5</td>
</tr>
<tr>
<td>G</td>
<td>Put up the exterior siding</td>
<td>D</td>
<td>7</td>
</tr>
<tr>
<td>H</td>
<td>Do the exterior painting</td>
<td>E/G</td>
<td>9</td>
</tr>
<tr>
<td>I</td>
<td>Do the electrical work</td>
<td>C</td>
<td>7</td>
</tr>
<tr>
<td>J</td>
<td>Put up the wallboard</td>
<td>F/I</td>
<td>3</td>
</tr>
<tr>
<td>K</td>
<td>Install the flooring</td>
<td>J</td>
<td>4</td>
</tr>
<tr>
<td>L</td>
<td>Do the Interior painting</td>
<td>J</td>
<td>5</td>
</tr>
<tr>
<td>M</td>
<td>Install the exterior fixtures</td>
<td>H</td>
<td>2</td>
</tr>
<tr>
<td>N</td>
<td>Install the interior fixtures</td>
<td>K/L</td>
<td>6</td>
</tr>
</tbody>
</table>

You can use an AON network (activity at the node) to represent the project, with precedence relationships represented by the arrows (directed edges):
The duration for each activity is indicated for each vertex.

Two-term activities 0 are added to the list: the beginning and end of the project.

A Project Manager aims to answer the following questions:

1. What is the time required to execute the project if there are delays?
2. When is the latest can the individual activities begin and end in order to meet the deadline?
3. The sooner the work can begin and end without causing delays?
4. What are the bottleneck activities where any delay affect the time limit?
5. For other activities, what is the tolerable delay without affecting the run?

To answer these questions we use a two-pass procedure: a direct passage that calculates the earliest start (ES) and end earlier (EF) of each vertex, beginning with the first, and a reverse pass that calculates the later start (LS) and end later (LF) of each activity, starting with the end:

**Direct passage**

Activity “beginning”, ES and EF = 0 = 0

The activity, ES and EF = 0 = 2, can not have an end sooner than the duration of the activity; B activity,

It is = 2 is EF = 6 Can not start before the previous activity has ended; whether an activity is longer than a previous one, can not begin before the end of that longer it takes, for example
H activity have to wait until both G and E activities have ended, logo IS It must be equal to the largest EF the immediately preceding activities, then ES = 29

The procedure continues until the project lifetime (longer path).

With the direct procedure are calculated all earlier possible date (first chance schedule)

Transient reverse activity “order”, LF = 44 and LS = 44

M activity, you must start no later than LS = 44 - 2 = 42 and LF = 44

H activity, LF = LS de M = 42 it has to end for the M have its start later and LS = 42 - 9:

With the reverse procedure are calculated all possible dates later (last chance schedule).

With these procedures each vertex of the network gets a set of information:

that allow to analyze the impact of delays and possible gaps throughout the network activities:

The clearance of activity is the difference between the end later and end earlier:

\[ \text{Backlash} = \text{LF} - \text{EF} \]

Note that could be used later start and earlier start for \( \text{LF} - \text{EF} = \text{LS} - \text{ES} \)

The activities with zero slack belong to the critical path in the case of this example:

PERT

Planning with limited resources

Rather than adopt the probability distribution for the duration, it is common to use three estimates:

Optimistic duration: \( o \)

Duration probabilistic: \( m \)

Duration pessimistic: \( p \)

where \( o \leq m \leq p \).

The model is then based on the beta distribution

where the expected value (mean) \( \mu = (o + 4m + p)/6 \) is and the variance is \( \sigma^2 = (p - o)^2 / 36 \).

Then the CPM is carried out with the expected duration and the results found is the expected duration of the project.

This method of finding the CPM is designated by PERT (Program Evaluation and Review Technique) and the method is often referred to as PERT / CPM.

There are specific software for the management of projects such as MS Project, Wrike, Basecamp, Bitrix24, Zoho and many others.

Many of these software allow to enter the activities, duration, relationship and the clearances directly to a table called Gantt chart (Gantt Henry, 1910).
If the duration is a random variable, the values are used: optimistic, pessimistic and likely duration of each activity and the beta distribution to calculate the average and variance:

In addition to planning problems and time management (time costs), we may be faced with the problem of economic costs activities, budgeting and cost control of the project.

The planning problem assumes that there are infinite resources to the activities being performed, but in practice we have to introduce the limited resources, which makes the LP-hard problem.

For example, the activity can be performed on the two units of time, with 3 resources in a professional category and 5 units, which are: A (2, 2, 5).

In principle, the formulation of a planning problem with limited resources adds to the planning problem formulation more restrictions on the use of available resources.

Thus the restriction equations lose unimodular property and the problem becomes a full schedule, it will not be treated here.

**Elements of Game theory**

**Introduction**

This activity studies the modeling of decision-making problems involving conflict of interests by intervening.

One classic example is the zero-sum game between two players, where one side wins and the other loses and cooperation between the players does not bring any benefit to all.

**Activity Details**

**Game Theory**

In game theory studying the cases of decision-making involving conflict of interests by intervening:

Game of 2 players (no draws)

Diplomatic or political conflict

Business competition

Competitive biological systems

Just as an introduction to game theory, study the following example:

Two persons A and B, each with a coin, simultaneously show a side of the coin to the opponent:

The win if both show costly (11) or both show crowns (00)

B wins if a show face and the other shows the crown (10) or vice versa (01).

The prospect of the winning or losing, the payment matrix (pay-off matrix) is as follows (in the
The attempt to maximize the minimum profit that B grants you - Maximin strategy A
B tries to minimize as much prejudice that A wants to inflict - MiniMax strategy B
If a player is more likely to win (because it is more experienced, or stronger, or is doped) so the game will not be zero gain.

Example
Consider the following Player payment matrix A

- Player A has two strategies and Player B has three strategies.
- To identify the dominant strategies and saddle point payment table given:
- The maximum values of columns are: (max column)
- The minimum values of the lines (min row)
As there are equal amounts of maxcolumn and minrow, then there are also dominant strategies: The player should always play the 2 and player B strategy should always play the strategy 3.
The entry in row 3 and column 2 is called a saddle-point return table given.

Exemple
Consider the following Player payment matrix A

- Player A has two strategies and plans to maximize the minimum profit that B grants you:
- Player A wants to know the frequency with which p1 should play strategy 1 and p2 frequency that should play strategy 2.
- Player B also has two strategies and aims to minimize as much prejudice that A wants to inflict:
- Player B wants to know the frequency with which q1 should play strategy 1 and the frequency with which q2 should play strategy 2.

First we identify the dominant strategies and point saddle given payment table:

- We are the maximum values of columns (max column)
- The minimum values of the lines (min row)
• as there are no equal values max column and min row, also there are no dominant strategies.

For values of \( p_1, p_2 \) and \( q_1, q_2 \) Calculate up the difference between the values of module lines and the difference module and the column values are added; the proportion of values in rows and columns with the total is the value you want:

The expected value of the income is \( 3p_1 + p_2 = 3(1/6) + (5/6) = 8/6 \)

The expected value of the losses is \( 3q_1 + 2q_2 = 3(2/6) - 2(4/6) = 8/6 \)

The reason this result has to do with the following:

for player A

B is consistently play then strategy 1 the advantage of A is \( 3p_1 + p_2 \) and B consistently play strategy 2 then the advantage of A is \( -2p_1 + 2p_2 \);

Player A plan to use Maximin

maximize \( u \)

with constraints (minimization of expressions \( p_1 \))

\[
\begin{align*}
    u &\leq 3p_1 + p_2 \\
    u &\leq -2p_1 + 2p_2
\end{align*}
\]

if no saddle point then solving \( 3p_1 + p_2 = -2p_1 + 2p_2 \) simultaneously with \( p_1 + p_2 = 1 \)

is obtained \( p_1 = 1/6, p_2 = 5/6 \)

to player B

The play is then consistently strategy 1 the injury B is \( 3q_1 + 2q_2 \). The consistently and then throw the second strategy the injury B is \( q_1 + 2q_2 \);

Player B intends to use the MiniMax:

minimize \( v \)

with restrictions (maximizing expressions \( q_1 \))

\[
\begin{align*}
    3q_1 + 2q_2 &\geq v \\
    q_1 + 2q_2 &\geq v
\end{align*}
\]

if no saddle point then, by solving simultaneously taking account that \( q_1 + q_2 = 1 \) is obtained \( q_1 = 2/6, q_2 = 4/6 \)

The problem can also be solved by the graphical method:

the two expressions for the benefit of A are given by

\[
\begin{align*}
    3p_1 + p_2 \\
    -2p_1 + 2p_2
\end{align*}
\]

but how \( p_1, p_2 \) They are related by \( p_1 + p_2 = 1 \), the intersection of the two graphs the
results desired by the iminponto max

Example

Consider the following payment matrix for player A.

There are no common values between the maxcolumn and minrow.

So there is no saddle point.

The advantage of A when B consistently apply the strategies 1, 2 and 3 respectively, is given by the expressions:

\[
\begin{align*}
3p_1 &- 2p_2 \\
4p_1 &- 3p_2 \\
p_1 &+ 3p_2
\end{align*}
\]

taking into account the probabilistic equation: \( p_1 + p_2 + p_3 = 1 \)

Using the graphical method:

get the values \( p_1 = 5/7 \) e \( p_2 = 2/7 \) and the expected value of the income is

\( p_1 + 3p_2 = (5/7) + 3(2/7) = 11/7 \)

Obtaining the values \( q_1, q_2, q_3 \) can be obtained also in this way (but it’s easier using the formulation of the PPL).

The problem of optimizing the game strategy can be formulated as a linear programming problem. The primal formulation for player A is given by:

Maximize

Restrictions

\[
\begin{align*}
u &\leq 3p_1 & - 2p_2 \\
u &\leq 4p_1 & - 3p_2 \\
u &\leq p_1 & + 3p_2 \\
p_1 & + p_2 & = 1 \\
p_1, p_2 & & \geq 0
\end{align*}
\]

unrestricted worth

The primal formulation for player B is given by:

Minimize \( v \)

Restrictions

\[
\begin{align*}
v &\geq 3q_1 & + 4q_2 & + q_3 \\
v &\geq -2q_1 & + 3q_2 & + 3q_3 \\
q_1 & + q_2 & + q_3 & = 1 \\
q_1, q_2 & & \geq q_3 & \geq 0
\end{align*}
\]
unrestricted worth If we calculate the dual formulation for player B by applying the conversion rules:

**Dual variables:**

\[
\begin{align*}
y_1 & : 3y_1 + 4q_2 + q_3 v \leq 0 \\
y_2 & : -2q_1 - 3y_2 + 3q_3 v \leq 0 \\
w & : q_1 + q_2 + q_3 = 1
\end{align*}
\]

Minimize \( w \)

**Restrictions**

\[
\begin{align*}
3y_1 + 2y_2 + w & \geq 0 \\
4y_1 + 3y_2 + w & \geq 0 \\
y_1 + 3y_2 + w & \geq 0
\end{align*}
\]

with

\[
y_1, y_2 \geq 0
\]

\( w \) unrestricted in sign.

it turns out that is the primal problem of player A, with the variables \( y_1, y_2, w \) renamed to \( p_1, p_2, u \).

It can be seen also that the dual problem of player A is the primal B.

Using SOLVER the primal Player A, we obtain:

the values \( p_1 = 0.71 \) \( p_2 = 0.25 \) and the expected value of the income is the same value, equal to those obtained above.

Using the solver to solve the primal Player B, we obtain the values of :

\[
\begin{align*}
q_1 & = 0.29 = 2/7, q_2 = 0, q_3 = 0.71 = 5/7.
\end{align*}
\]

**Conclusion**

Applications of cases study of the methods studied were presented in the previous units, in planning projects and in game theory.

**Assignment**

Exercise 1:

You need to take a trip by car to another town that you have never visited before. Therefore, you are studying a map to determine the shortest route to your destination. Depending on
which route you choose, there are five other towns (call them A, B, C, D, E) that you might pass through on the way. The map shows the mileage along each road that directly connects two towns without any intervening towns. These numbers are summarized in the following table, where a dash indicates that there is no road directly connecting these two towns without going through any other towns.

<table>
<thead>
<tr>
<th>Town</th>
<th>Miles between adjacent towns</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
</tr>
<tr>
<td>Origin</td>
<td>40</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
</tr>
</tbody>
</table>

Formulate this problem as a shortest-path problem by drawing

A. network where nodes represent towns, links represent roads, and numbers indicate the length of each link in miles.

B. Use the shortest path algorithm to solve this problem.

C. Formulate and solve a spreadsheet model for this problem.

D. If each number in the table represented your cost (in dollars) for driving your car from one town to the next, would the answer in part (b) or (c) now give your minimum cost route?

E. If each number in the table represented your time (in minutes) for driving your car from one town to the next, would the answer in part (b) or (c) now give your minimum time route?
Exercise 2:

Suppose that England, France, and Spain produce all the wheat, barley, and oats in the world. The world demand for wheat requires 125 million acres of land devoted to wheat production. Similarly, 60 million acres of land are required for barley and 75 million acres of land for oats. The total amount of land available for these purposes in England, France, and Spain is 70 million acres, 110 million acres, and 80 million acres, respectively. The number of hours of labor needed in England, France and Spain to produce an acre of wheat is 118, 13, and 16, respectively. The number of hours of labor needed in England, France, and Spain to produce an acre of barley is 15, 12, and 12, respectively. The number of hours of labor needed in England, France, and Spain to produce an acre of oats is 12, 10, and 16, respectively. The labor cost per hour in producing wheat is $9.00, $7.20, and $9.90 in England, France, and Spain, respectively. The labor cost per hour in producing barley is $8.10, $9.00, and $8.40 in England, France, and Spain respectively. The labor cost per hour in producing oats is $6.90, $7.50, and $6.30 in England, France, and Spain, respectively. The problem is to allocate land use in each country so as to meet the world food requirement and minimize the total labor cost.

(a) Formulate this problem as a transportation problem by constructing the appropriate parameter table.

(b) Draw the network representation of this problem.

(c) Obtain an optimal solution.

Unit Summary

This unit were presented network flow optimization problem.

Also have been through the basics and terminology of stochastic processes Markov;

A small introduction to Markov processes and developed the theory of queues was made to a queue and a server.

Were finally you present applications and some case studies, namely the problem of project planning and game theory.

Unit Readings and Other Resources

The readings in this unit are to be found at course level readings and other resources.

Module summaryhttps://docs.google.com/document/d/17XzMgvZpawyRyOP464yE7l87RsYK1N K8zWMPMo-xUsI/edit - heading=h.44sinio
Course Assessment

Final Examination

Instruction

This is a Two hours exam. No electronic calculator is allowed. Please justify your answer properly. The case study is to be submitted 24 hours later. Questions 2,3 and 4 are to be done under supervision

Case Study: CUTTING CAFETERIA COSTS

A cafeteria at All-State University has one special dish it serves like clockwork every Thursday at noon. This supposedly tasty dish is a casserole that contains sautéed onions, boiled sliced potatoes, green beans, and cream of mushroom soup. Unfortunately, students fail to see the special quality of this dish, and they loathingly refer to it as the Killer Casserole. The students reluctantly eat the casserole, however, because the cafeteria provides only a limited selection of dishes for Thursday’s lunch (namely, the casserole). Maria Gonzalez, the cafeteria manager, is looking to cut costs for the coming year, and she believes that one sure way to cut costs is to buy less expensive and perhaps lower-quality ingredients. Because the casserole is a weekly staple of the cafeteria menu, she concludes that if she can cut costs on the ingredients purchased for the casserole, she can significantly reduce overall cafeteria operating costs. She therefore decides to invest time in determining how to minimize the costs of the casserole while maintaining nutritional and taste requirements.

Maria focuses on reducing the costs of the two main ingredients in the casserole, the potatoes and green beans. These two ingredients are responsible for the greatest costs, nutritional content, and taste of the dish. Maria buys the potatoes and green beans from a wholesaler each week. Potatoes cost $0.40 per pound, and green beans cost $1.00 per pound. All-State University has established nutritional requirements that each main dish of the cafeteria must meet. Specifically, the total amount of the dish prepared for all the students for one meal must contain 180 grams (g) of protein, 80 milligrams (mg) of iron, and 1,050 mg of vitamin C. (There are 453.6 g in 1 lb and 1,000 mg in 1 g.) For simplicity when planning, Maria assumes that only the potatoes and green beans contribute to the nutritional content of the casserole. Because Maria works at a cutting-edge technological university, she has been exposed to the numerous resources on the World Wide Web. She decides to surf the Web to find the nutritional content of potatoes and green beans. Her research yields the following nutritional information about the two ingredients: Potatoes Green Beans Protein 1.5 g per 100 g 5.67 g per 10 ounces Iron 0.3 mg per 100 g 3.402 mg per 10 ounces Vitamin C 12 mg per 100 g 28.35 mg per 10 ounces (There are 28.35 g in 1 ounce.) Edson Branner, the cafeteria cook who is surprisingly concerned about taste, informs Maria that an edible casserole must contain at least a six to five ratio in the weight of potatoes to green beans. Given the number of students who eat in the cafeteria, Maria knows that she must purchase enough potatoes and green beans to prepare a minimum of 10 kilograms (kg) of casserole each week. (There are 1,000 g in 1 kg.)
Again for simplicity in planning, she assumes that only the potatoes and green beans determine the amount of casserole that can be prepared. Maria does not establish an upper limit on the amount of casserole to prepare, since she knows all leftovers can be served for many days thereafter or can be used creatively in preparing other dishes.

(a) Determine the amount of potatoes and green beans Maria should purchase each week for the casserole to minimize the ingredient costs while meeting nutritional, taste, and demand requirements. Before she makes her final decision, Maria plans to explore the following questions independently except where otherwise indicated.

(b) Maria is not very concerned about the taste of the casserole; she is only concerned about meeting nutritional requirements and cutting costs. She therefore forces Edson to change the recipe to allow for only at least a one to two ratio in the weight of potatoes to green beans. Given the new recipe, determine the amount of potatoes and green beans Maria should purchase each week.

(c) Maria decides to lower the iron requirement to 65 mg since she determines that the other ingredients, such as the onions and cream of mushroom soup, also provide iron. Determine the amount of potatoes and green beans Maria should purchase each week given this new iron requirement.

(d) Maria learns that the wholesaler has a surplus of green beans and is therefore selling the green beans for a lower price of $0.50 per lb. Using the same iron requirement from part (c) and the new price of green beans, determine the amount of potatoes and green beans Maria should purchase each week.

(e) Maria decides that she wants to purchase lima beans instead of green beans since lima beans are less expensive and provide a greater amount of protein and iron than green beans. Maria again wields her absolute power and forces Edson to change the recipe to include lima beans instead of green beans. Maria knows she can purchase lima beans for $0.60 per lb from the wholesaler. She also knows that lima beans contain 22.68 g of protein per 10 ounces of lima beans, 6.804 mg of iron per 10 ounces of lima beans, and no vitamin C. Using the new cost and nutritional content of lima beans, determine the amount of potatoes and lima beans Maria should purchase each week to minimize the ingredient costs while meeting nutritional, taste, and demand requirements. The nutritional requirements include the reduced iron requirement from part (c).

(f) Will Edson be happy with the solution in part (e)? Why or why not? (g) An All-State student task force meets during Body Awareness Week and determines that All-State University’s nutritional requirements for iron are too lax and that those for vitamin C are too stringent. The task force urges the university to adopt a policy that requires each serving of an entrée to contain at least 120 mg of iron and at least 500 mg of vitamin C. Using potatoes and lima beans as the ingredients for the dish and using the new nutritional requirements, determine the amount of potatoes and lima beans Maria should purchase each week.

2. (20%) Consider the following problem

\[
\text{Max} \quad 3x_1 + 2x_2
\]
Subject to

\[3x_1 + x_2 \leq 12\]
\[x_1 + x_2 \leq 6\]
\[5x_1 + x_2 \leq 27\]
\[x_1, x_2 \geq 0\]

a. Solve the problem by the simplex method and identify the complementary basic solution for the dual problem at each iteration.

b. Write the dual problem, and then solve the problem by the dual simplex method. Compare the resulting sequence of basic solutions with the complementary basic solutions obtained in part a).

3. (30%) Consider the following problem

Max \( 40x_1 + 50x_2 + 60x_3 \)

Subject to

\[x_1 + 5x_2 + x_3 \leq 10\]
\[3x_1 + 2x_2 + 4x_3 \leq 60\]
\[2x_2 \leq 24\]
\[x_1, x_2, x_3 \geq 0\]

with the optimal tableau as follows:

<table>
<thead>
<tr>
<th></th>
<th>(z)</th>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(x_3)</th>
<th>(x_4)</th>
<th>(x_5)</th>
<th>(x_6)</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>(z)</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>35/2</td>
<td>15/2</td>
<td>0</td>
<td>975</td>
</tr>
<tr>
<td>(x_1)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>-1/2</td>
<td>1/2</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>(x_6)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>-3/2</td>
<td>1/2</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>(x_2)</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1/2</td>
<td>3/4</td>
<td>-1/4</td>
<td>0</td>
<td>15/2</td>
</tr>
</tbody>
</table>

a. Constraint 1 is a constraint on available machine hours. For a cost of $20, I can work additional overtime hours. Should I?

b. R&D has just come up with a new product, which is represented by variable 7, with \(c_7 = 20\) and \(a_{17} = 1\), \(a_{27} = 2\) and \(a_{37} = 2\). Should I make this product?

c. What is the optimal solution if \(a_{13}\) changes from 2 to -1

d. What is the optimal solution if \(b_{13}\) changes from 30 to 10
e. What is the optimal solution change if $a_{12}$ changes from 2 to 5?

f. What is the optimal solution if the constraint $x_1 + 2x_2 + x_3 \leq 29$ is added?

**Grading scheme**

The total assignment weigh 50% 10% for the case study and 40% for the remaining two questions questions.

**References**


Operations Research, Prof. G. Srinivasan, NPTEL, 2009;

http://www.me.utexas.edu/~jensen/exercises/mst_spt/mst_spt.html

Academia Khan - https://khanacademy.org/

Linear and Non-Linear Programming, 3rd Ed, David Luenberger, Yinyu Ye, Springer, 2008;


http://www.mhhe.com/engcs/industrial/hillier/splash/

International Federation of Operational Research Societies - http://ifors.org/web/

The African Virtual University Headquarters

Cape Office Park
Ring Road Kilimani
PO Box 2v5405-00603
Nairobi, Kenya
Tel: +254 20 25283333
contact@avu.org
oer@avu.org

The African Virtual University Regional Office in Dakar

Université Virtuelle Africaine
Bureau Régional de l’Afrique de l’Ouest
Sicap Liberté VI Extension
Villa No.8 VDN
B.P. 50609 Dakar, Sénégal
Tel: +221 338670324
bureauregional@avu.org