NWU®

FACULTY OF ENGINEERING

GROUP SEMESTER ASSIGNMENT

REII 313 – Object Oriented programming

Submitted to: Mr. R Luies

Date: 11/06/2020

Author: J.G. Roos 29870089

Table of Contents

Contents

List of Figures

1. Introduction

As described by the assignment information given by our lecturer an individual or a group of two students was expected to write an application that could solve any given Linear Programming problem. This report gives feedback on the application software developed. The software is intended to use the Simplex method and branch and bound calculations. However, more will be discussed in this report. The aim of the application is to make calculations of linear problems easier, addressing the Linear Problem (Simplex) and Integer Problem (Branch and Bound).

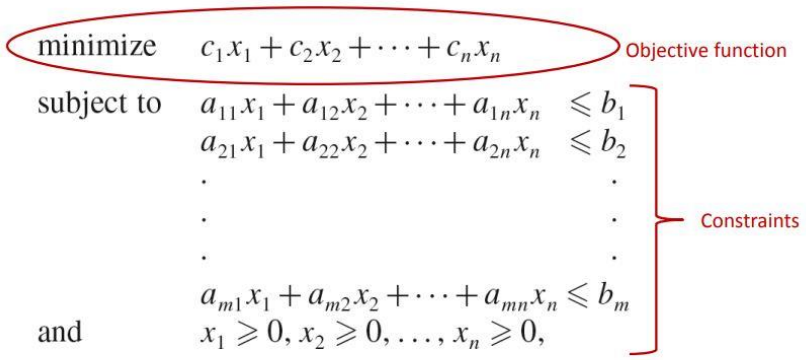
2.1 Background

We live in a world filled with finite resources, like time and money, and the goal is not to waste the resources given to use but use it the best possible way, getting the most out of every resource. “Optimization is a way of living” [1] , Linear programming is one of the most simplistic ways of optimizing. “Linear Programming Problems are concerned with the efficient use or allocation of limited resources to meet desired objectives.” [2]

The aim was to use two linear programming techniques in our solver, to explain Linear programming in more detail we will be using these two techniques.

2.2.1 Simplex Method

The Simplex Method can be described as an iterative method improves the solution after each “run”. The process stops when it is not possible to improve the solution anymore. The goal of the function is to maximize or minimize the objective function based on the user’s desire.



$c_1, c_2, \ldots, c_n$ are the cost coefficients
$x_1, x_2, \ldots, x_n$ are the decision variables (the unknowns to be determined)
$a_{11}, a_{12}, \ldots, a_{mn}$ are the constraint coefficients
$b_1, b_2, \ldots, b_m$ are the constraint requirements (right-hand-side values)

FIGURE 1: COMPONENTS OF SIMPLEX

As you will notice in the Figure 1 above the simplex method is designed to work any number of variables subjected to any number of constraints. The simplex makes use of slack variables and Matrices do get to the optimal solution.

2.2.2 Branch and Bound

The branch and bound is typically used to solve Integer programming problems, however it is not limited to only solving these types of problems. Therefore, it could be described as a solution approach that can be applied to a number of different types of problems. It works on the principle divide and conquer, the solutions are broken down into smaller subsets and are evaluated until the optimal solution given Certain parameters are found.

To best illustrate the branch and bound method in term of calculations and programming will be to use Figure 2. Figure 2 illustrates a tree data structure. The first "circle" you will see represents the Root node, the two nodes underneath will be the children nodes, making the root node also a parent node. The nodes contain information of the variables, upper-bound (UB) and lower-bound(LB). How to find this info:

- The UB is found by calculating the best solutions Linearly (in terms of our project using Simplex) for the number of variables, x_1, x_2, …., x_n, given all the restrictions and information, and applying this info to the "Cost function" to find the maximum possible solution.
- LB is found rounding down the values of x_1, x_2, …., x_n, found with the Linear programming calculation, to the nearest possible integer and calculating the "Cost Function" accordingly.

The process there after is to see which variable is furthest from the previous or next possible integer and use the rounded up and rounded down integers to add a new parameter. (For example $x_1 = 5.6$ and $x_2 = 1.3$, we will round x1 up and down to find the parameters of our new nodes). The idea is the to compare branches coming from the parent node and see which delivers the higher value inside the UB and LB of the root node. This is done till the highest possible solution is found with only integers.

2. Application and Program Overview

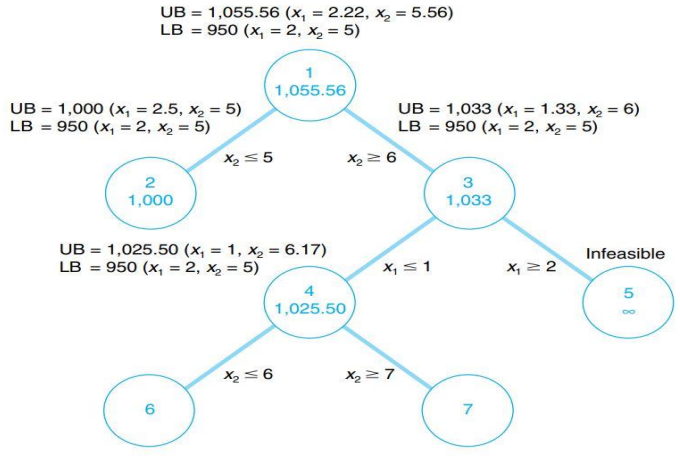


FIGURE 2: TREE

3.1 GUI Interface

The GUI or graphic user interface is a window where the user can interact with the program. The GUI is set up in such a way to get information from the user. This is information

regarding the Linear problem. When running the application, the first window will open (see Figure 3 below):

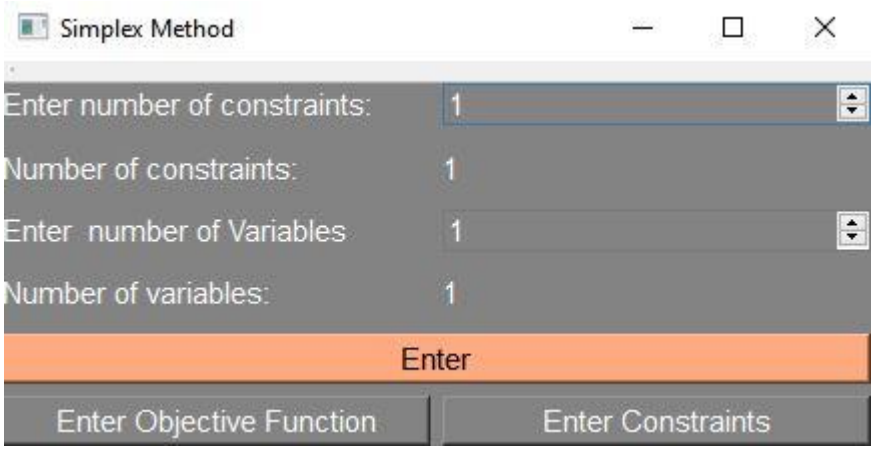


FIGURE 3: LINEAR PROBLEM INFORMATION (VARIABLES AND CONSTRAINTS).

In this window you will notice that there are two integer "spin boxes" this allows the user to set the number of constraints and the number of variables in the Linear problem. Both boxes have a minimum of 1 and maximum of 1000, however it is unlikely that a user will have 1000 constraints and variables. If these two factors are set the user will press enter, this will alter the numbers below the spinboxes to show the user the program noted the decision. After pressing the enter key the user should press either the "Objective Function" "PushButton" or the "Enter constraints" "PushButton".

When pressing the "Enter Objective Function" button a pop up window, illustrated by Figure 4, will appear allowing the user to enter the weights of the variables of the objective function. The window will appear for the number of variables the user has selected. When the user has entered the value (once again with the use of a spin box) they will press the "Ok" "PushButton", the window will automatically close after the number of variables have been inserted.

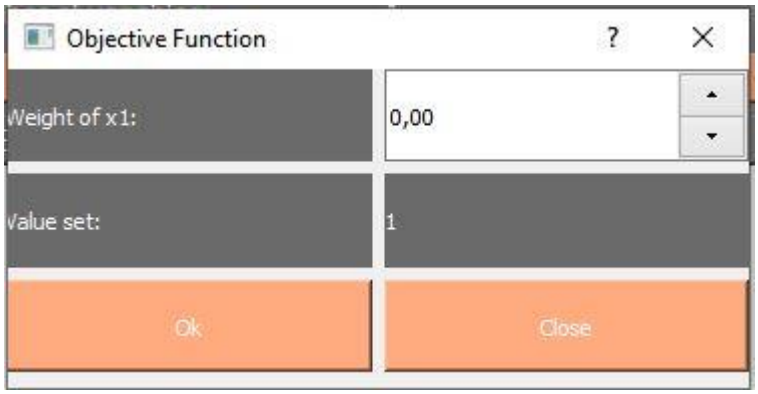


FIGURE 4: OBJECTIVE FUNCTION POP-UP WINDOW

When pushing the "Enter Constraints" button the Constraints window will open up giving the user the opportunity to enter the weights of the variables in each equation.

FIGURE 5: CONSTRAINTS WINDOW

3.2 Limitations

Regarding the project:

The biggest limitation regarding my project was time. Me and my former team member were on track to submit when a few hiccups arose, causing the group to split, deciding each member should do the project on their own. This made it hard to draw up a simplex method in time.

Regarding the program:

The program can take the information, however there is no program to do the calculations.

3.3 UML

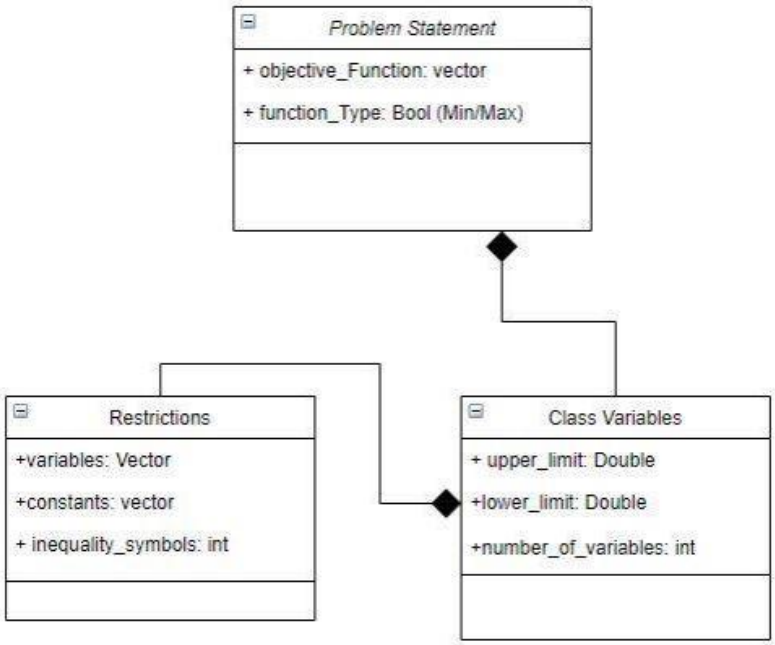


FIGURE 6: UML (IN THEORY)

Figure 6 shows the UML that we planned to work with, after looking deeper into a solver system the UML might look a bit more intense in response to the actual code and functions that needs to be used. Unfortunately, there is now practical UML based on actual code due to the lack there-off.

3.4 Example

3. Conclusion

A solver is a very good application to have, especially when it comes to Industrial Engineers where optimization plays such a large part. Linear problems could get very difficult to do on paper, not impossible, but errors could easily be made. Mistakes are humane, therefore taking the human aspect out of calculations ensures you'll get always get the right answer that will be easy to test.

4. References

[1] S. R. Gass SI, Linear Programming Methods of apllication, New York: Dover Publications, Inc., 2003.

[2] Anonymous, "Analytics Vidhya," 26 February 2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/02/lintroductory-guide-on-linear-programming-explained-in-simple-english/.