**CCT College Dublin**

**Assessment Cover Page**

|  |  |
| --- | --- |
| **Module Title:** | Artificial Intelligence  Data Visualisation & Comms |
| **Assessment Title:** | Homework #2 |
| **Lecturer Name:** | David McQuaid  Sam Weiss |
| **Student Full Name:** | Yuri Mendonca Ribeiro |
| **Student Number:** | 2020347 |
| **Assessment Due Date:** | 05/01/2024 |
| **Date of Submission:** | 10/01/2024 |

**Declaration**

|  |
| --- |
| By submitting this assessment, I confirm that I have read the CCT policy on Academic Misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source. I declare it to be my own work and that all material from third parties has been appropriately referenced. I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution. |

Contents

[Artificial Intelligence 3](#_Toc155820727)

[Data Visualisation 6](#_Toc155820728)

[GITHUB 8](#_Toc155820729)

[Bibliography 8](#_Toc155820730)

# Artificial Intelligence

For these tasks, Scenario 1 and Scenario 2, I opted to CSP using the Google OR-TOOLS library. Having the variables, in this case, the candidates for a new position in Ciara’s start-up it was given a Boolean value 0 for not selected and 1 for selected which is called domain. After the variables were set up the constraints for each Scenario.

First, we have the relation between people and abilities:

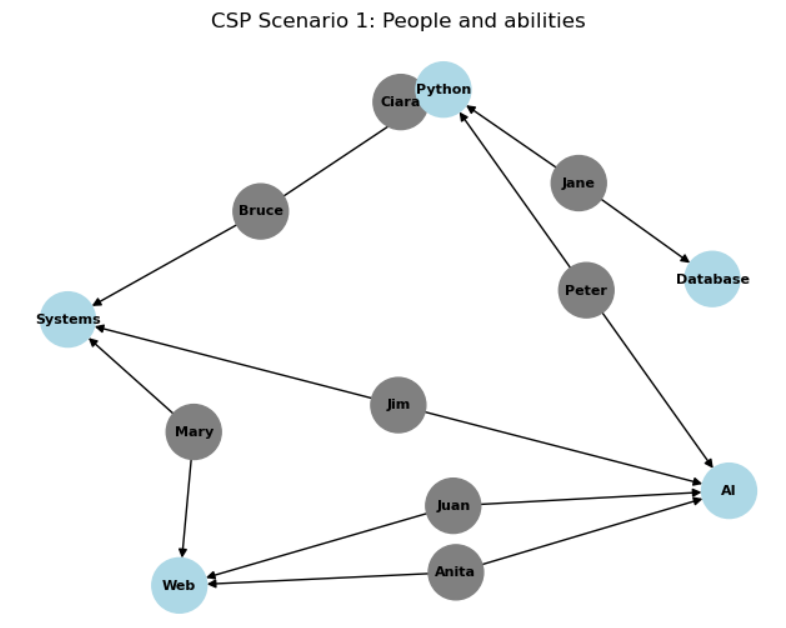


Figure - People and abilities

For scenario 1 the constraints are:

* Only 3 people can be hired besides Ciara
* If a person has two abilities, he/she can take two roles in the company
* All roles must be fulfilled

The result is: The first combination is Ciara (Python) + Jane (Python and Database), Jim (AI and Systems) and Juan (Web and AI) following the second combination Ciara (Python) + Anita (Web and AI), Jane (Python and Database) and Jim (AI and Systems).

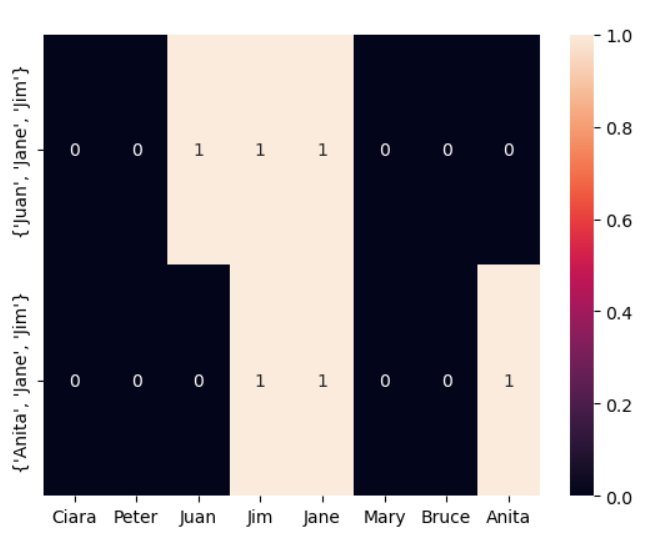


Figure - Result Scenario #1

For scenario 2 the constraints are:

* Only 4 people can be hired besides Ciara and Juan
* If a person has two abilities, he/she can take two roles in the company
* All roles must be fulfilled

The result is: The first combination is Ciara (Python), Juan (Web and AI) + Anita (Web and AI), Jane (Python and Database), Jim (AI and Systems) and Peter (Python and AI) following the second combination Ciara (Python), Juan (Web and AI) + Jane (Python and Database), Jim (AI and Systems) and Peter (Python and AI).

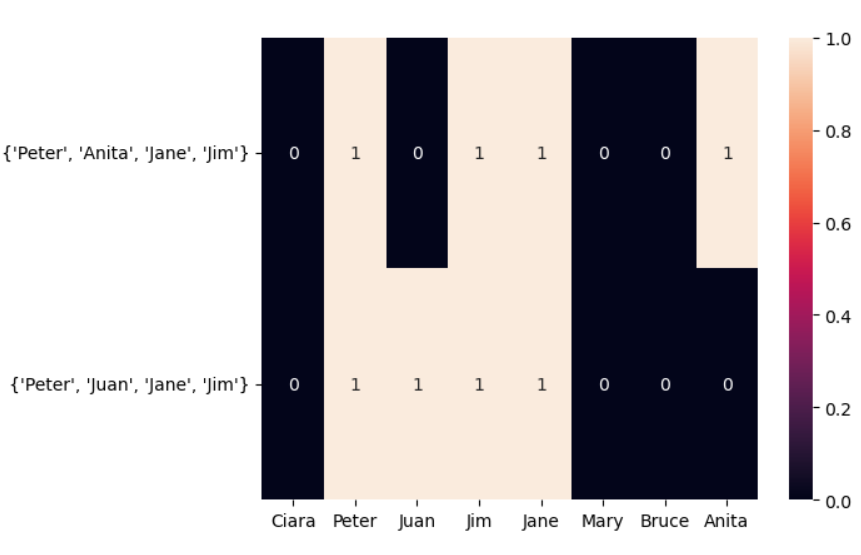


Figure - Result Scenario #2

The Constraint Satisfaction Problem differs from standard searching algorithms when it comes to the order, in which we set the variables. In a CSP the order is irrelevant whereas for a shortest path algorithm, the order does make the difference and can have an impact on the solution of the problem, not reflecting the best solution. (Simic, 2022)

The second difference is how classical algorithms deliver the solutions. They do not inspect their inner structures while CSPs know the individual variables making the solution and set them one by one. (Simic, 2022)

Using a simple Depth-First Search (DFS) without the satisfies\_constraints function, the algorithm cannot find the correct combination based on the restrictions. The result is all the combinations with 3 people, as shown in the code.

# Data Visualisation

For a better understanding of the results created by the algorithm, I have chosen a heatmap for its intuitive visualisation, enhancing the exploration user experience.

First I created a binary matrix where 1 represents the match of the person and the ability required in both scenarios. The Seaborn library was used for the heatmap creation where the x-axis is the people and the y-axis is the unique combination. The colour used is the default colour scheme of Seaborn.

For the Interactive visualisations, I have opted to use Plotly library to create an interactive heatmap. After converting the binary combinations into a data frame, I used Plotly Express to create the heatmap with interactive features such as labels with the results appearing when the cursor of the mouse is pointed to any part of the heatmap where the colour yellow represents the match of the person with the combination in contrast of the purple colour (Viridis scale).

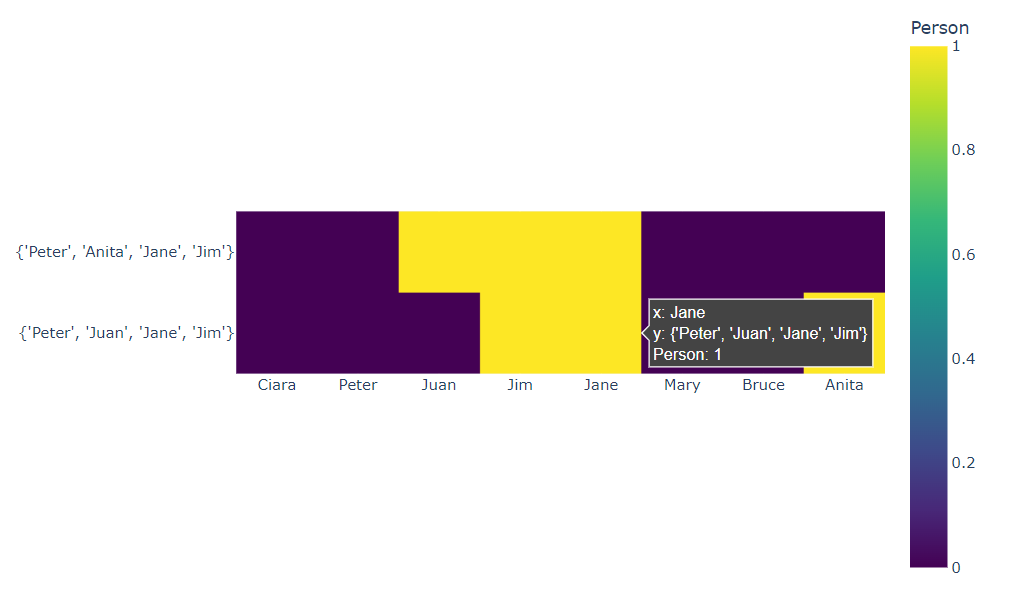


Figure - Interactive Visualisation

In the heatmap represented in Figure 4 above, we can easily identify the two combinations possible:

1 – Peter, Anita, Jane and Jim

2 – Peter, Juan, Jane and Jim

Represented by the yellow colour and the binary scale on the left from 0 - 1, where 1 represents the match between the people and the abilities required.

After get the right combination I created a simple GUI where anyone can be able to try different combinations to see if it matches the requirements:

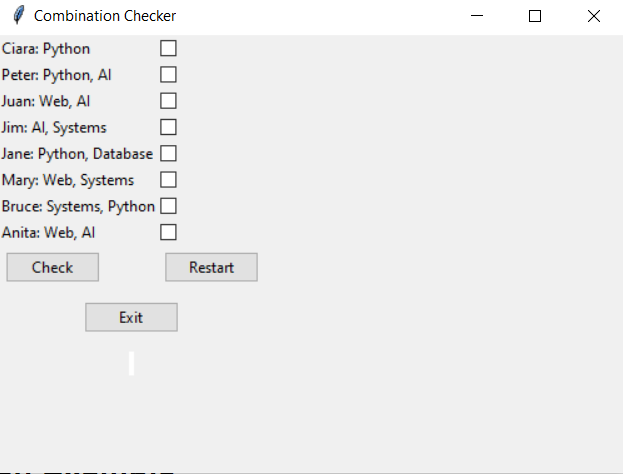


Figure - GUI - Combination Checker

For example, if I check the first combination of the scenario 1: Juan, Jim and Jane

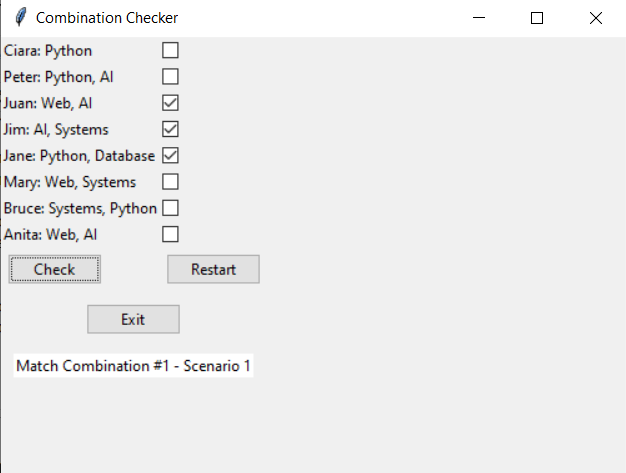


Figure - GUI - Match Combination #1 - Scenario 1

However, if I check a random selection the Combination Checker will return an error message:

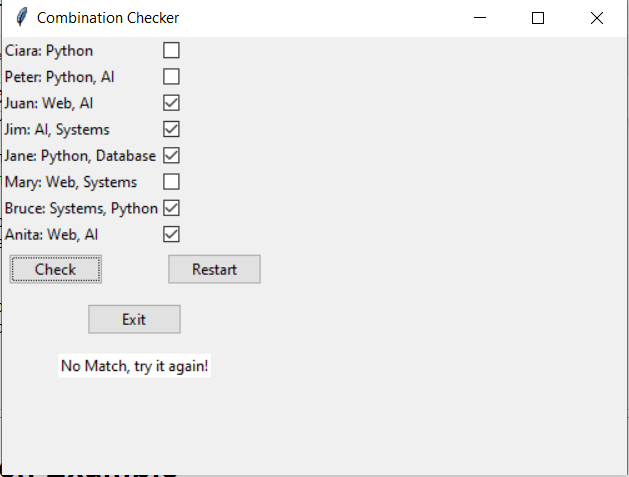


Figure - GUI - Combination Checker - No Match

# GITHUB

Link for the code: https://github.com/YWRCCT/YuriRibeiro\_AI\_DV\_-ICA

# Bibliography

Simic, M. (2022, February 10). *How to Solve Constraint Satisfaction Problems*. Retrieved from Baeldung: https://www.baeldung.com/cs/csp

[Figure 1- People and abilities 3](#_Toc155820709)

[Figure 2- Result Scenario #1 4](#_Toc155820710)

[Figure 3- Result Scenario #2 5](#_Toc155820711)

[Figure 4- Interactive Visualisation 6](#_Toc155820712)

[Figure 5- GUI - Combination Checker 7](#_Toc155820713)

[Figure 6 - GUI - Match Combination #1 - Scenario 1 7](#_Toc155820714)

[Figure 7- GUI - Combination Checker - No Match 8](#_Toc155820715)