KINGDOM OF SAUDI ARABIA
Ministry of Education
UNIVERSITY OF JEDDAH
Main Campus- Girl's
COLLEGE OF COMPUTER SCIENCE & ENGINEERING
COMPUTER SCIENCE & AI DEPARTMENT

جامعة جدة
University of Jeddah

المملكة العربية السعودية
وزارة التعليم
جـــامـعة جدة
شطر الطالبات
كلية علوم وهندسة الحاسب
قسم علوم الحاسب والذكاء الاصطناعي

Constraint Satisfaction Problems

CCAI 221 project

A2L

Students	ID
Arjwan Alharbi	2110826
Ruba Alsulami	2110618
Teif Saleh	2111370
Raghad Alamri	2110102
Joud Farhan	2112233

CSP Problem:

In charge of scheduling for computer science classes that meet weekly. There are 5 classes
that meet and 3 professors who will be teaching these classes. We are constrained by the fact that
each professor can only teach one class at a time.
The classes are:

1. Class 1 - Intro to Programming: meets from 8:00-9:00am
2. Class 2 - Intro to Artificial Intelligence: meets from 8:30-9:30am
3. Class 3 - Natural Language Processing: meets from 9:00-10:00am
4. Class 4 - Computer Vision: meets from 9:00-10:00am
5. Class 5 - Machine Learning: meets from 10:30-11:30am

The professors are:

1. Professor A, who is qualified to teach Classes 1, 2, and 5.
2. Professor B, who is qualified to teach Classes 3, 4, and 5.
3. Professor C, who is qualified to teach Classes 1, 3, and 4.

Domain:

Each variable in the CSP will represent a class and its corresponding assigned professor. The domain of each variable will be the set of all possible professors who are qualified to teach that particular class.

Variables:

The variables in this CSP will be the five classes.

Class 1: Intro to Programming

Class 2: Intro to Artificial Intelligence

Class 3: Natural Language Processing

Class 4: Computer Vision

Class 5: Machine Learning

Unary Constraints:

Unary constraints are restrictions on individual variables. In this case, unary constraints would be the time slots in which each class takes place. The unary constraints can be represented as follows:

Class 1: [8:00-9:00am]

Class 2: [8:30-9:30am]

Class 3: [9:00-10:00am]

Class 4: [9:00-10:00am]

Class 5: [10:30-11:30am]

Binary Constraints:

Binary constraints are restrictions between two variables. In this case, binary constraints would be the qualifications of each professor. The binary constraints can be represented as follows:

Professor A: [Class 1, Class 2, Class 5]

Professor B: [Class 3, Class 4, Class 5]

Professor C: [Class 1, Class 3, Class 4]

Node-Consistency:

Node-consistency is a consistency checking algorithm that checks each node (variable) against its domain and constraints. After enforcing node-consistency, the domains of the variables would look like this:

Class 1: [Professor A, Professor C]

Class 2: [Professor A]

Class 3: [Professor B, Professor C]

Class 4: [Professor B, Professor C]

Class 5: [Professor A, Professor B]

Arc-Consistency:

Arc-consistency is a consistency checking algorithm that checks each arc (constraint) in the constraint graph to ensure that there are no inconsistencies between the variables. After enforcing arc-consistency, the domains of the variables would look like this:

Class 1: [Professor A, Professor C]

Class 2: [Professor A]

Class 3: [Professor B, Professor C]

Class 4: [Professor B, Professor C]

Class 5: [Professor A, Professor B]

Here, the domains of the variables remain the same after enforcing node-consistency and arc-consistency. This means that the CSP is already consistent, and no more restrictions need to be applied.

Explanation of Code:

·The code implements the backtracking algorithm to find a solution to the CSP problem defined by the variables, domains, unary constraints, and binary constraints.
·The function backtracking takes two arguments: assignment and variables. The assignment argument is a dictionary that keeps track of the current assignments of variables, and the variables argument is a list of all variables.
·The function is_consistent checks if the current assignment is consistent with the constraints defined in the problem.
·If all variables have been assigned, the function returns the assignment.
·If there are unassigned variables, the function chooses an unassigned variable and tries each value in its domain.
·If the assignment is consistent with constraints, the function continues with the next variable.
·If the assignment leads to a contradiction, the function backtracks by deleting the current assignment and trying the next value in the domain.
·The function returns None if no solution is found.
·Finally, the code calls the backtracking function and prints the solution if one is found, otherwise it prints "No solution found".

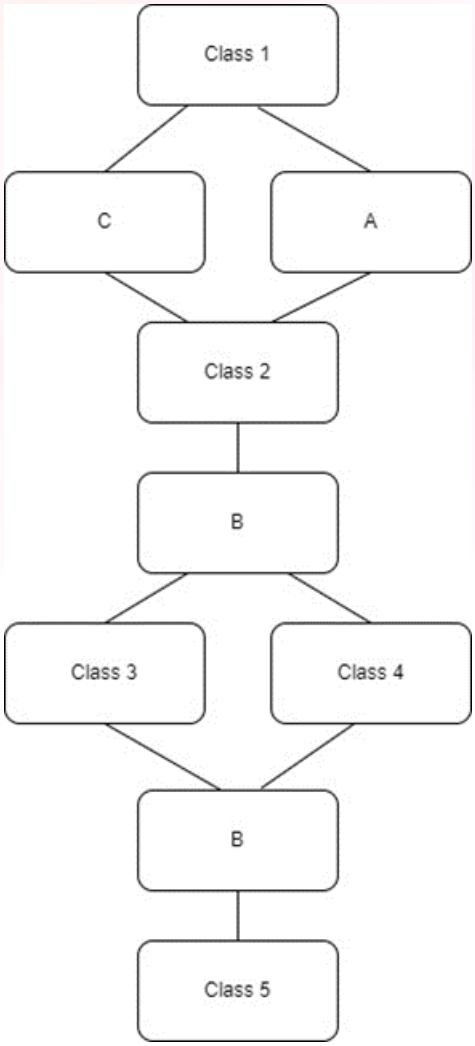
Output:

```
Solution: {'Class 1': 'Professor C', 'Class 2': 'Professor A', 'Class 3': 'Professor B', 'Class 4': 'Professor C', 'C
lass 5': 'Professor A'}
```

:

Constraint Graph:

A constraint graph is a graphical representation of the CSP, where each node represents a variable and each arc represents a constraint between two variables. The constraint graph for this CSP would look like this:



References

https://freecontent.manning.com/constraint-satisfaction-problems-in-python/

http://aima.cs.berkeley.edu/python/csp.py

https://edge.edx.org/c4x/BerkeleyX/CS188-FA14/asset/section_4_solutions.pdf