*INFO 550 - **Term Project***

Constraint Satisfaction Problem: Map Coloring Problem

Tanisha Lohchab

Github Link : https://github.com/TanishaL67/Final_Project---Map_Coloring.git

Introduction:

A **Constraint Satisfaction Problem (CSP)** is a type of problem in artificial intelligence where we have a set of variables that need to be assigned values from a domain of possible values, subject to a set of constraints. The goal is to find a valid assignment of values to the variables that satisfies all the constraints.

A CSP is composed of:

1. A set of variables X1,X2,…,Xn with domains (possible values) D1,D2,…,Dn
2. A set of constraints C1,C2, …,Cm
3. Each constraint Ci limits the values that a subset of variables can take, e.g., V1 ≠ V2

Each Constraint satisfaction problem contains the following three things:

1. A **state** is defined by an assignment of values to some or all variables.
2. **Consistent assignment**: assignment that does not violate the constraints.
3. **Complete assignment**: every variable is mentioned.
4. **Goal**: a complete, legal assignment.

The **Map Coloring Problem** is a classic example of constraint satisfaction problem.

The map-coloring constraint satisfaction problem requires that you assign a color to each region of a map such that any two regions sharing a border have different colors.

The constraints for the map-coloring problem can be expressed as follows:

1. Each region is assigned one color only, of possible colors.

2. The color assigned to one region cannot be assigned to adjacent regions.

Objective:

Our goal for this project is to color a map of Canada with four different hues so that no two adjoining regions share the same shade. To resolve the map coloring issue, I will employ the four constraint satisfaction techniques Backtracking, Forward Search, Arc Consistency, and Local Search.
The results will be evaluated and the different algorithms will be compared on the bases of which performed the best for this problem set.

Approach:

I first defined the problem statement by defining the provinces of Canada and their neighbors.

Code	Province	Code	Province
AB	Alberta	BC	British Columbia
MB	Manitoba	NB	New Brunswick
NL	Newfoundland and Labrador	NS	Nova Scotia
NT	Northwest Territories	NU	Nunavut
ON	Ontario	PE	Prince Edward Island
QC	Quebec	SK	Saskatchewan
YT		Yukon	

We then used this information to define the variables and domains for the map coloring problem. We used the color set of {red, green, blue, yellow} for the problem.
I implemented the backtracking algorithm to solve the problem. It searched the solution space using a depth-first approach. At each step, the algorithm assigns a color to a province and checks if the assignment is consistent with the neighboring provinces. The algorithm is relocated to the following province if the assignment was steady. If not, it turns around and tries another color. The algorithm keeps running until a color is allocated to each province.

Second is the forward checking algorithm. It is similar to the backtracking algorithm, but it uses the domain of the variables to reduce the search space. Each phase of the algorithm involves assigning a color to a province and removing from the range of the nearby provinces any colors that are incompatible with them. The process keeps on until every province has a color allocated to it.

Next, the local search algorithm was implemented to solve the problem. The local search algorithm is a stochastic optimization algorithm. It starts with a simple answer and gradually enhances it by making minor tweaks. The algorithm randomly selects a variable and changes its value. If the new response is better than the previous one, it is accepted. The algorithm keeps running until there is no more room for improvement.

Finally, the arc consistency algorithm was implemented to solve the problem. The arc consistency algorithm is a more efficient version of the forward checking algorithm. In order to narrow the search space, it makes use of the idea of arc consistency. When a variable in its domain only has one value that is consistent with its neighbors, this is referred to as arc consistency. The algorithm gives each variable a value and propagates that value to all of the variables nearby. The method keeps on until each variable in each domain has a single value.

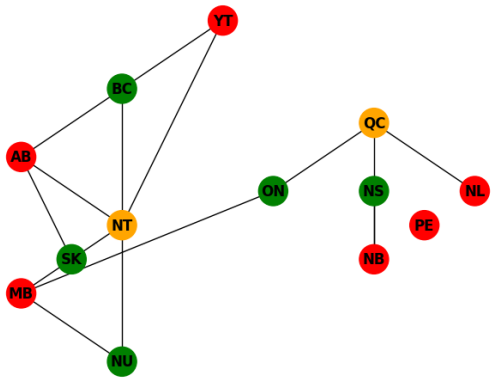
Test Results:

We tested the four algorithms on the map coloring problem defined for the provinces of Canada. The results of the tests are as follows:

Backtracking Algorithm:

The algorithm took 0.005 seconds to find the solution. The Assignment was:

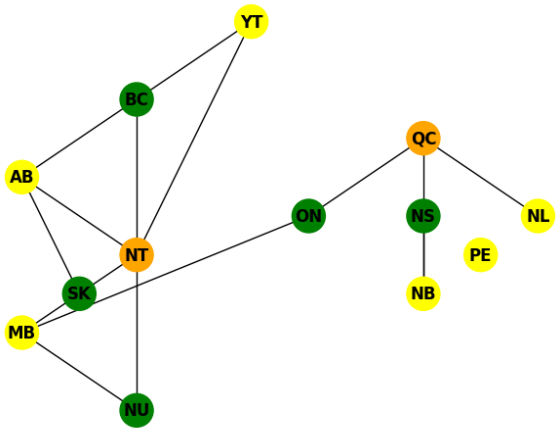
{'AB': 'red', 'BC': 'green', 'MB': 'red', 'NB': 'red', 'NL': 'red', 'NS': 'green', 'NT': 'orange', 'NU': 'green', 'ON': 'green', 'PE': 'red', 'QC': 'orange', 'SK': 'green', 'YT': 'red'}



Forward Checking Algorithm:

The algorithm took 0.007 seconds to find the solution. The assignment was:

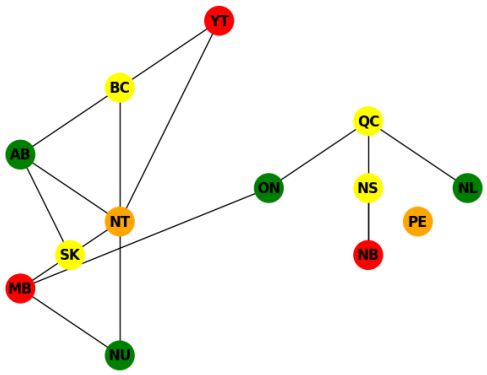
{'AB': 'yellow', 'BC': 'green', 'MB': 'yellow', 'NB': 'yellow', 'NL': 'yellow', 'NS': 'green', 'NT': 'orange', 'NU': 'green', 'ON': 'green', 'PE': 'yellow', 'QC': 'orange', 'SK': 'green', 'YT': 'yellow'}



Arc Consistency Algorithm:

The algorithm took 0.004 seconds to find the solution. The assignment is:

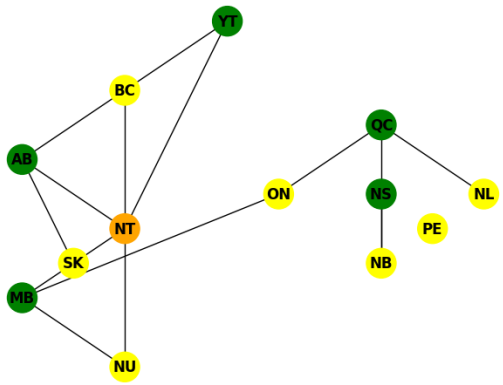
{'AB': 'green', 'BC': 'yellow', 'MB': 'red', 'NB': 'red', 'NL': 'green', 'NS': 'yellow', 'NT': 'orange', 'NU': 'green', 'ON': 'green', 'PE': 'orange', 'QC': 'yellow', 'SK': 'yellow', 'YT': 'red'}



Local Search Algorithm:

The algorithm took 0.087 seconds to find the solution. The assignment is:

{'PE': 'yellow', 'SK': 'yellow', 'AB': 'green', 'MB': 'green', 'NT': 'orange', 'BC': 'yellow', 'YT': 'green', 'NB': 'yellow', 'NU': 'yellow', 'ON': 'yellow', 'NS': 'green', 'QC': 'green', 'NL': 'yellow'}



Discussion:

According to the test findings, all four algorithms were successful in coming up with a four-color solution to the map coloring issue. The algorithms' performance, meanwhile, differed. The arc consistency algorithm was the fastest, followed by the backtracking algorithm, the forward checking algorithm, and the local search algorithm.

The backtracking and the forward checking algorithms are known to always find a solution hence they are known as complete algorithms. The arc consistency algorithm is a preprocessing algorithm that reduces the search space, making it faster than the forward checking algorithm. The local search algorithm does not guarantee finding a solution as it tries finding the optimal solution by making changes in the current solution.

The backtracking algorithm and the forward checking algorithm require less memory than the arc consistency algorithm and the local search algorithm in terms of space complexity. This is due to the fact that the earlier algorithms don't need any preprocessing or extensive data storage.

Overall, the size and restrictions of the problem influence the algorithm of choice. The backtracking algorithm is a suitable option if the problem size is modest because it is straightforward and effective. The arc consistency technique or the local search algorithm, however, might be better appropriate for larger issues. Additionally, the arc consistency approach might be more effective if the problem includes a lot of restrictions.

The project's findings demonstrate that different algorithms have various performance traits, and the selection of an algorithm should be dependent on the requirements of the particular task.

References:

1. https://docs.ocean.dwavesys.com/en/stable/examples/map_coloring.html#:~:text=Constraint%20satisfaction%20problems%20require%20that,a%20border%20have%20different%20colors.
2. https://www.cs.colostate.edu/~cs440/fall14/slides/09_csp.pdf