



Intelligent Systems

Project-3

**Constraint satisfaction problems (CSP) - Map
Coloring**

Project Report

Srinivasulu Padigay (801330017)

(Note: This is a solo project)

Table of Contents

- 1. Introduction**
- 2. Heuristics**
 - a. Minimum remaining values (MRV)**
 - b. Degree Constraint**
 - c. Least Constraining Value**
- 3. File and Program Structure**
 - a. File Structure**
 - b. Program Structure**
 - c. Description of the Programming Part**
 - d. Description of Typical System Customers**
- 4. Execution Output Screenshots**
 - a. Screenshots**
- 5. Results in Tabular format**
- 6. Conclusion**

1. Introduction

Exploring Map Coloring with Constraint Satisfaction Techniques

In the fascinating realm of computational theory and artificial intelligence, Constraint Satisfaction Problems (CSP) represent a unique and versatile class of problems. This project delves into the intriguing world of CSP through the lens of map coloring, a classical example that beautifully illustrates the principles and challenges inherent in CSPs.

Map coloring is a problem that can be visualized as assigning colors to various regions (such as countries on a map) in such a way that no adjacent regions share the same color. This seemingly simple task embodies the essence of constraint satisfaction, where each coloring choice must adhere to specific rules or constraints. The project specifically focuses on k-coloring, where 'k' denotes the number of available colors. The challenge lies in finding the minimum value of 'k' (known as the chromatic number, $\chi(G)$) that allows for a feasible coloring of the map, ensuring that adjacent countries or regions are not colored the same.

Our journey begins with the exploration of map coloring techniques applied to the maps of the USA and Australia. These two diverse geographical entities present unique challenges and opportunities to investigate the nuances of CSP. The core objective is to compute the chromatic number for both maps and to explore various search methodologies in solving the CSP of map coloring.

Methodologies Employed:

The project will employ the following methods, initially without the use of heuristics, to explore different dimensions of the problem:

- 1. Depth First Search (DFS) Only:** This approach involves a simple, unguided depth-first traversal of the solution space.
- 2. DFS + Forward Checking:** Enhancing DFS with forward checking to prune the search space by eliminating choices that violate constraints.
- 3. DFS + Forward Checking + Propagation through Singleton Domains:** An advanced method where propagation through singleton domains is utilized alongside DFS and forward checking to further optimize the search process.

In addition to these non-heuristic methods, the project also aims to implement heuristic-based approaches to enhance the efficiency and effectiveness of the solution. These heuristics include:

- **Minimum Remaining Values (MRV)**
- **Degree Constraint**
- **Least Constraining Value**

Each heuristic offers a unique strategy in selecting the next variable or value, aiming to reduce the complexity and improve the performance of the search algorithms.

Experimentation and Analysis:

The project will conduct experiments on both the USA and Australia maps, using the aforementioned methods with and without heuristics. The key parameters to be observed and analyzed include:

- The number of backtracks encountered.
- The time required to compute the results.
- The solution itself, particularly in terms of the chromatic number achieved.

By contrasting the results obtained from both maps and across different methodologies, this project aims to provide insightful conclusions about the effectiveness of various CSP techniques in map coloring and potentially offer a comparative analysis of heuristic vs. non-heuristic approaches in solving CSPs.

This comprehensive exploration will not only deepen our understanding of CSPs in theoretical and practical aspects but also shed light on the potential applications and optimizations possible in this fascinating domain of artificial intelligence and computational problem-solving.

2. Heuristics

Understanding Heuristics in Constraint Satisfaction Problems: A Deep Dive with the Australia Map Coloring Example

Introduction to Heuristics in CSP

In the realm of Constraint Satisfaction Problems (CSP), heuristics play a pivotal role in enhancing the efficiency of search algorithms. These heuristics are essentially strategies or rules of thumb that guide the search process towards a solution in a more efficient manner compared to a blind or brute-force search. In the context of map coloring, and specifically with the Australia map as our case study, the use of heuristics can significantly streamline the process of finding the minimum number of colors needed (the chromatic number) while ensuring that no two adjacent regions share the same color.

The Three Key Heuristics Explored

Minimum Remaining Values (MRV): This heuristic prioritizes the choice of the next region (or node) to color based on the number of remaining legal color choices available. The region with the fewest legal colors left (i.e., the most constrained one) is selected first. This approach tends to reduce the likelihood of encountering a dead-end in the search process.

- **Application on Australia Map:** Consider Australia's map divided into its states and territories. Using MRV, we would first color the state or territory that has the least number of legal coloring options remaining. For instance, if South Australia, which borders many other territories, has fewer color options left compared to other states, it would be prioritized for coloring.

Degree Constraint: This heuristic involves selecting the next region to color based on how many other regions it constrains, i.e., the one with the most neighbors. The rationale is that by coloring the most connected region earlier, you reduce the complexity of subsequent choices.

- **Application on Australia Map:** In this case, a state like New South Wales, which shares borders with several other states, might be chosen earlier in the coloring process because coloring it impacts the coloring choices of multiple neighboring states.

Least Constraining Value: This heuristic is about choosing the color that will leave the maximum number of options open for the adjacent regions. It aims to minimize the immediate impact of the current decision on future choices.

- **Application on Australia Map:** When selecting a color for, say, Queensland, the chosen color would be the one that restricts the fewest color options for its neighboring states like New South Wales and South Australia.

Integrating Heuristics in the Australia Map Coloring

In practice, these heuristics can be combined for a more efficient search strategy. For the Australia map coloring, the process would involve first selecting a state using the MRV or Degree Constraint heuristic. Once a state is chosen, the Least Constraining Value heuristic would come into play to decide the color to be assigned to that state.

The effectiveness of these heuristics can be quantitatively measured by observing the number of backtracks needed during the search process and the time taken to reach a solution. In theory, the application of these heuristics should result in fewer backtracks and a quicker discovery of the optimal solution compared to methods that do not utilize heuristics.

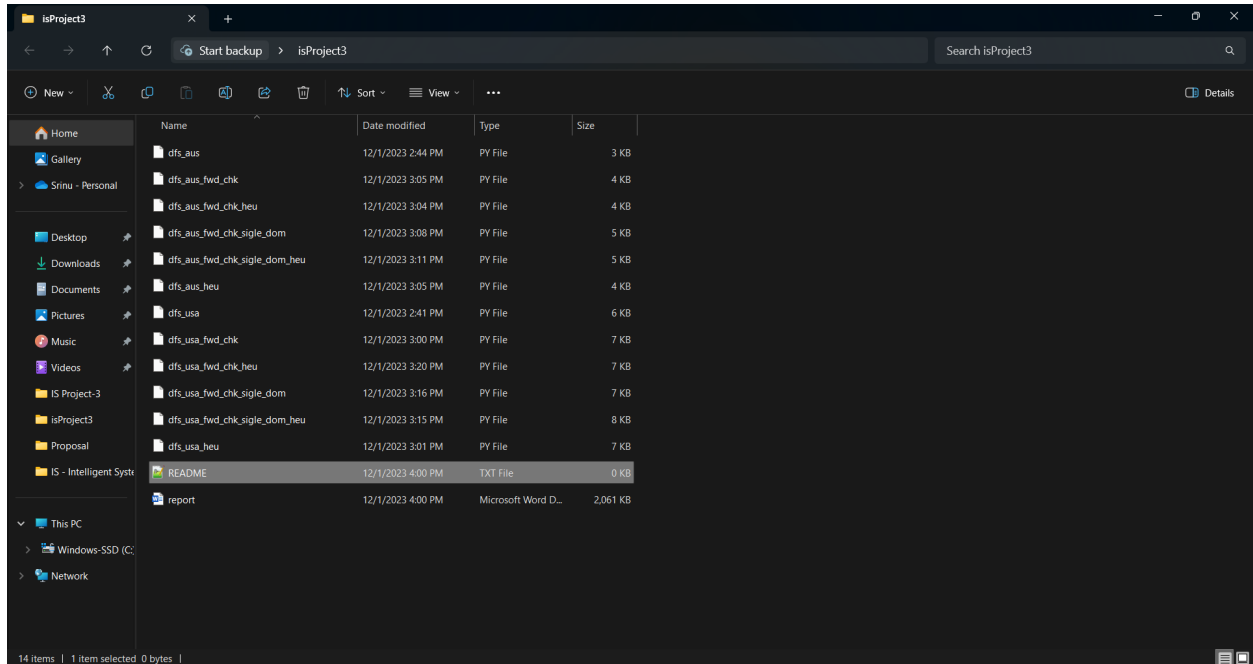
3. File and Program Structure

3.a File Structure

All python programs are standalone programs that execute independently and produce results.

Report is a word document explaining the project and code.

ReadMe is a text file explaining how to run the code.



3.b Program Structure

1. dfs_australia.py

- **Problem Formulation:** This script solves the map coloring problem for Australia using Depth-First Search (DFS).
- **Global Variables:**
 - **australia_map:** A dictionary representing the map of Australia with adjacency information for all states and territories.
 - **colors:** A list of available colors.
- **Key Functions:**
 - **is_valid_color:** Checks if a color can be assigned to a state without violating constraints.
 - **dfs_coloring:** Implements DFS to find a valid coloring of the map.

- **next_state:** Determines the next state to consider in the DFS.
- **map_coloring:** The main function that initiates the DFS coloring process.

2. dfs_aus_fwd_chk.py

- **Enhancement:** Adds forward checking to the DFS approach.
- **Key Functions:**
 - **forward_check:** Checks if assigning a color to a state leaves viable options for its neighbors.
 - **restore_colors:** Restores the available colors for the neighbors of a state.

3. dfs_aus_fwd_chk_heu.py

- **Enhancement:** Incorporates heuristics along with forward checking.
- **Heuristic Functions:**
 - **mr_selection:** Selects the state with the minimum remaining values (MRV) heuristic.
 - **degree_constraint:** Chooses a state based on the degree heuristic.
 - **lc_order:** Orders colors based on the least constraining value (LCV) heuristic.

4. dfs_aus_fwd_chk_sigle_dom.py

- **Enhancement:** Implements forward checking with propagation through singleton domains.
- **Key Function:**
 - **propagate_singleton_domains:** Propagates constraints when a state has only one remaining color option.

5. dfs_aus_fwd_chk_sigle_dom_heu.py

- **Enhancement:** Combines forward checking, propagation through singleton domains, and heuristics.
- **Key Functions:**
 - **select_unassigned_variable:** Chooses the next state to assign a color using MRV and degree constraints.
 - **order_domain_values:** Orders the colors using the LCV heuristic.
 - **forward_check_and_propagate:** Performs forward checking and propagates constraints through singleton domains.

6. dfs_aus_heu.py

- **Enhancement:** Implements DFS with heuristics for the Australian map.
- **Heuristic Functions:**

- **mr_v_selection, degree_constraint, lcv_order:** Same as previously described.
- **is_valid_assignment:** Checks if a color assignment is valid.

7. dfs_usa.py

- **Problem Formulation:** Solves the map coloring problem for the USA using DFS.
- **Global Variables:** Similar to dfs_aus.py, but for the USA map.
- **Key Functions:** Similar structure to dfs_aus.py.

8. dfs_usa_fwd_chk.py

- **Enhancement:** Implements forward checking in DFS for the USA map.
- **Key Functions:**
 - **is_valid_color:** Checks if a color can be assigned to a state without violating constraints.
 - **forward_check:** Ensures that assigning a color to a state leaves viable options for its neighbors.
 - **restore_colors:** Restores the available colors for the neighbors of a state.
 - **dfs_with_forward_checking:** The main DFS function that incorporates forward checking.

9. dfs_usa_fwd_chk_heu.py

- **Enhancement:** Combines DFS, forward checking, and heuristics for the USA map.
- **Heuristic Functions:**
 - **select_unassigned_variable:** Chooses the next state to assign a color using MRV and degree constraints.
 - **order_domain_values:** Orders the colors using the LCV heuristic.
 - **forward_check:** Performs forward checking after a color assignment.

10. dfs_usa_fwd_chk_sgle_dom.py

- **Enhancement:** Implements DFS with forward checking and propagation through singleton domains for the USA map.
- **Key Functions:**
 - **is_valid_color:** Checks if a color can be assigned to a state without violating constraints.
 - **forward_check:** Performs forward checking after a color assignment.
 - **restore_colors:** Restores the available colors for the neighbors of a state.
 - **dfs_with_forward_check_and_propagation:** The main DFS function that incorporates forward checking and propagation.

11. dfs_usa_fwd_chk_sgle_dom_heu.py

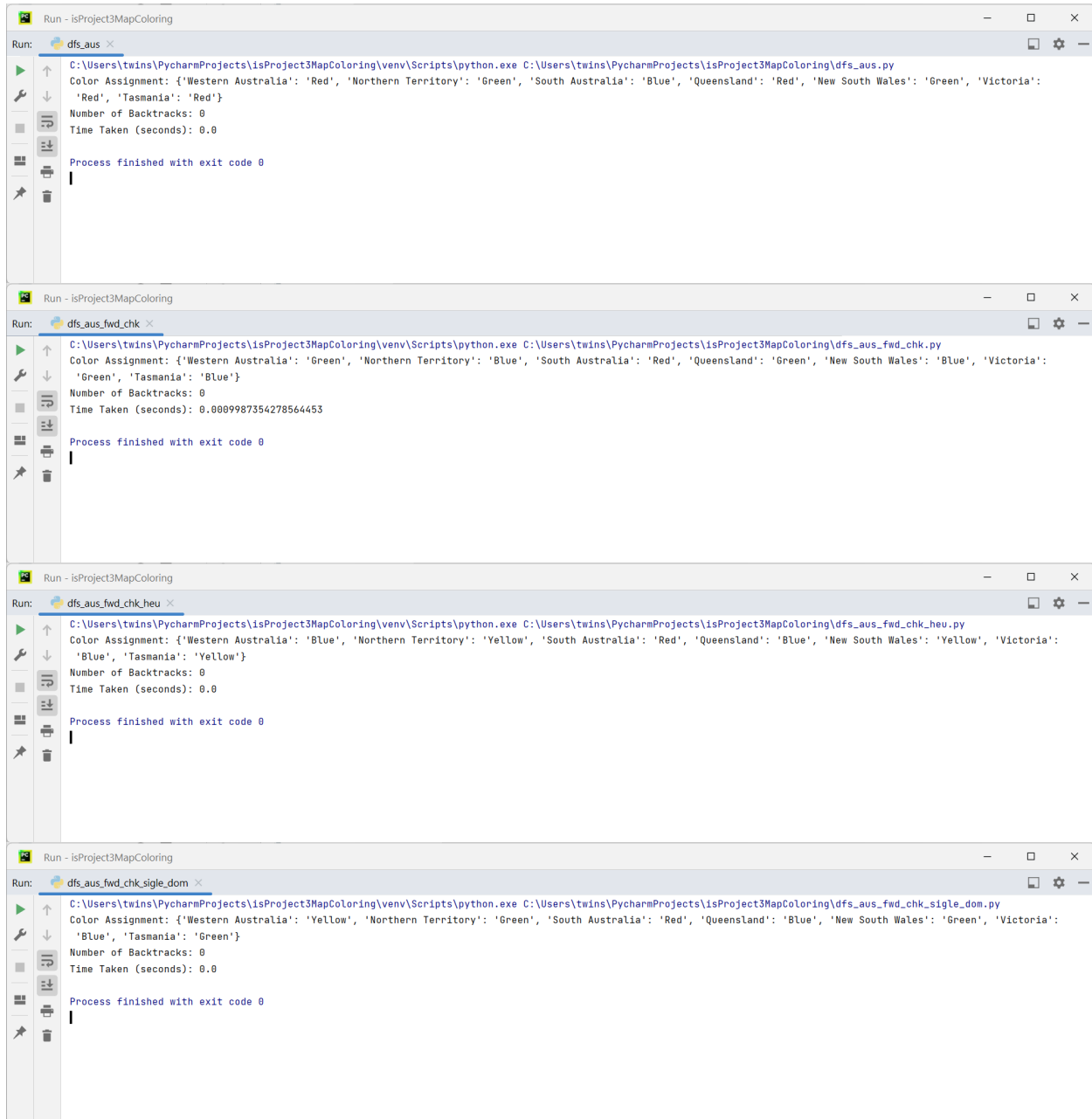
- **Enhancement:** Combines DFS, forward checking, propagation through singleton domains, and heuristics for the USA map.
- **Heuristic Functions:**
 - **select_unassigned_variable:** Chooses the next state to assign a color using MRV and degree constraints.
 - **order_domain_values:** Orders the colors using the LCV heuristic.
 - **forward_check_and_propagate:** Performs forward checking and propagates constraints through singleton domains.

12. dfs_usa_heu.py

- This is similar to dfs_aus_heu.py

4. Execution Output Screenshots

4.a Screenshots



The following screenshots show the execution output of four different DFS algorithms for map coloring, each in a PyCharm Run window titled "Run - isProject3MapColoring".

Run: dfs_aus

```
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_aus.py
Color Assignment: {'Western Australia': 'Red', 'Northern Territory': 'Green', 'South Australia': 'Blue', 'Queensland': 'Red', 'New South Wales': 'Green', 'Victoria': 'Red', 'Tasmania': 'Red'}
Number of Backtracks: 0
Time Taken (seconds): 0.0
Process finished with exit code 0
```

Run: dfs_aus_fwd_chk

```
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_aus_fwd_chk.py
Color Assignment: {'Western Australia': 'Green', 'Northern Territory': 'Blue', 'South Australia': 'Red', 'Queensland': 'Green', 'New South Wales': 'Blue', 'Victoria': 'Green', 'Tasmania': 'Blue'}
Number of Backtracks: 0
Time Taken (seconds): 0.0009987354278564453
Process finished with exit code 0
```

Run: dfs_aus_fwd_chk_heu

```
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_aus_fwd_chk_heu.py
Color Assignment: {'Western Australia': 'Blue', 'Northern Territory': 'Yellow', 'South Australia': 'Red', 'Queensland': 'Blue', 'New South Wales': 'Yellow', 'Victoria': 'Blue', 'Tasmania': 'Yellow'}
Number of Backtracks: 0
Time Taken (seconds): 0.0
Process finished with exit code 0
```

Run: dfs_aus_fwd_chk_sigle_dom

```
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_aus_fwd_chk_sigle_dom.py
Color Assignment: {'Western Australia': 'Yellow', 'Northern Territory': 'Green', 'South Australia': 'Red', 'Queensland': 'Blue', 'New South Wales': 'Green', 'Victoria': 'Blue', 'Tasmania': 'Green'}
Number of Backtracks: 0
Time Taken (seconds): 0.0
Process finished with exit code 0
```

```
Run - isProject3MapColoring
Run: dfs_aus_fwd_chk_sigle_dom_heu
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_aus_fwd_chk_sigle_dom_heu.py
Color Assignment: {'South Australia': 'Blue', 'Northern Territory': 'Red', 'Queensland': 'Green', 'New South Wales': 'Red', 'Victoria': 'Green', 'Western Australia': 'Green', 'Tasmania': 'Blue'}
Number of Backtracks: 0
Time Taken (seconds): 0.0

Process finished with exit code 0
```

```
Run - isProject3MapColoring
Run: dfs_aus_heu
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_aus_heu.py
Color Assignment: {'Western Australia': 'Green', 'Northern Territory': 'Yellow', 'South Australia': 'Blue', 'Queensland': 'Green', 'New South Wales': 'Yellow', 'Victoria': 'Green', 'Tasmania': 'Green'}
Number of Backtracks: 0
Time Taken (seconds): 0.0

Process finished with exit code 0
```

```
Run - isProject3MapColoring
Run: dfs_usa
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_usa.py
Color Assignment: {'Alabama': 'Red', 'Alaska': 'Red', 'Arizona': 'Red', 'Arkansas': 'Red', 'California': 'Green', 'Colorado': 'Green', 'Connecticut': 'Red', 'Delaware': 'Red', 'Florida': 'Green', 'Georgia': 'Blue', 'Hawaii': 'Red', 'Idaho': 'Red', 'Illinois': 'Red', 'Indiana': 'Green', 'Iowa': 'Green', 'Kansas': 'Red', 'Kentucky': 'Blue', 'Louisiana': 'Blue', 'Maine': 'Red', 'Maryland': 'Green', 'Massachusetts': 'Green', 'Michigan': 'Red', 'Minnesota': 'Blue', 'Mississippi': 'Yellow', 'Missouri': 'Yellow', 'Montana': 'Green', 'Nebraska': 'Blue', 'Nevada': 'Yellow', 'New Hampshire': 'Blue', 'New Jersey': 'Green', 'New Mexico': 'Yellow', 'New York': 'Yellow', 'North Carolina': 'Red', 'North Dakota': 'Yellow', 'Ohio': 'Yellow', 'Oklahoma': 'Blue', 'Oregon': 'Blue', 'Pennsylvania': 'Blue', 'Rhode Island': 'Blue', 'South Carolina': 'Green', 'South Dakota': 'Red', 'Tennessee': 'Green', 'Texas': 'Green', 'Utah': 'Blue', 'Vermont': 'Red', 'Virginia': 'Yellow', 'Washington': 'Green', 'West Virginia': 'Red', 'Wisconsin': 'Yellow', 'Wyoming': 'Yellow'}
Number of Backtracks: 4056429
Time Taken (seconds): 11.13138723373413

Process finished with exit code 0
```

```
Run - isProject3MapColoring
Run: dfs_usa_fwd_chk, dfs_usa_fwd_chk_heu
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_usa_fwd_chk_heu.py
Color Assignment: {'Missouri': 'Yellow', 'Tennessee': 'Blue', 'Kentucky': 'Green', 'Arkansas': 'Green', 'Oklahoma': 'Blue', 'Illinois': 'Blue', 'Iowa': 'Green', 'Nebraska': 'Blue', 'South Dakota': 'Yellow', 'Wyoming': 'Green', 'Colorado': 'Yellow', 'Utah': 'Blue', 'Idaho': 'Yellow', 'Nevada': 'Green', 'Virginia': 'Yellow', 'West Virginia': 'Blue', 'Ohio': 'Yellow', 'Indiana': 'Red', 'Pennsylvania': 'Green', 'Maryland': 'Red', 'Arizona': 'Yellow', 'Minnesota': 'Blue', 'Mississippi': 'Yellow', 'Alabama': 'Green', 'Georgia': 'Yellow', 'Oregon': 'Blue', 'California': 'Red', 'Texas': 'Yellow', 'Delaware': 'Yellow', 'Michigan': 'Blue', 'Montana': 'Blue', 'New Jersey': 'Blue', 'New York': 'Yellow', 'North Carolina': 'Green', 'Florida': 'Blue', 'Kansas': 'Green', 'Louisiana': 'Blue', 'New Mexico': 'Green', 'North Dakota': 'Green', 'South Carolina': 'Blue', 'Washington': 'Green', 'Wisconsin': 'Yellow', 'Massachusetts': 'Blue', 'Connecticut': 'Green', 'Vermont': 'Green', 'New Hampshire': 'Yellow', 'Rhode Island': 'Yellow', 'Maine': 'Blue', 'Alaska': 'Yellow', 'Hawaii': 'Yellow'}
Number of Backtracks: 0
Time Taken (seconds): 0.0020841465759277344

Process finished with exit code 0
```

```
Run - isProject3MapColoring
Run: dfs_usa_fwd_chk, dfs_usa_fwd_chk_sigle_dom
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_usa_fwd_chk_sigle_dom.py
Color Assignment: {'Alabama': 'Green', 'Alaska': 'Green', 'Arizona': 'Green', 'Arkansas': 'Green', 'California': 'Blue', 'Colorado': 'Green', 'Connecticut': 'Green', 'Delaware': 'Green', 'Florida': 'Blue', 'Georgia': 'Red', 'Hawaii': 'Green', 'Idaho': 'Green', 'Illinois': 'Green', 'Indiana': 'Blue', 'Iowa': 'Blue', 'Kansas': 'Blue', 'Kentucky': 'Red', 'Louisiana': 'Blue', 'Maine': 'Green', 'Maryland': 'Blue', 'Massachusetts': 'Blue', 'Michigan': 'Green', 'Minnesota': 'Green', 'Mississippi': 'Red', 'Missouri': 'Yellow', 'Montana': 'Red', 'Nebraska': 'Red', 'Nevada': 'Red', 'New Hampshire': 'Red', 'New Jersey': 'Blue', 'New Mexico': 'Blue', 'New York': 'Yellow', 'North Carolina': 'Green', 'North Dakota': 'Blue', 'Ohio': 'Yellow', 'Oklahoma': 'Red', 'Oregon': 'Yellow', 'Pennsylvania': 'Red', 'Rhode Island': 'Red', 'South Carolina': 'Blue', 'South Dakota': 'Yellow', 'Tennessee': 'Blue', 'Texas': 'Yellow', 'Utah': 'Yellow', 'Vermont': 'Green', 'Virginia': 'Yellow', 'Washington': 'Blue', 'West Virginia': 'Green', 'Wisconsin': 'Red', 'Wyoming': 'Blue'}
Number of Backtracks: 0
Time Taken (seconds): 0.0

Process finished with exit code 0
```

```
Run - isProject3MapColoring
dfs_usa_fwd_chk x dfs_usa_fwd_chk_sigle_dom_heu x
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_usa_fwd_chk_sigle_dom_heu.py
Color Assignment: {'Missouri': 'Green', 'Tennessee': 'Blue', 'Kentucky': 'Yellow', 'Arkansas': 'Yellow', 'Oklahoma': 'Blue', 'Illinois': 'Blue', 'Iowa': 'Yellow', 'Nebraska': 'Blue', 'South Dakota': 'Green', 'Wyoming': 'Yellow', 'Colorado': 'Green', 'Utah': 'Blue', 'Idaho': 'Green', 'Nevada': 'Yellow', 'Virginia': 'Green', 'West Virginia': 'Blue', 'Ohio': 'Green', 'Indiana': 'Red', 'Pennsylvania': 'Yellow', 'Maryland': 'Red', 'Arizona': 'Green', 'Minnesota': 'Blue', 'Mississippi': 'Green', 'Alabama': 'Yellow', 'Georgia': 'Green', 'Oregon': 'Blue', 'California': 'Red', 'Texas': 'Green', 'Delaware': 'Green', 'Michigan': 'Blue', 'Montana': 'Blue', 'New Jersey': 'Blue', 'New York': 'Green', 'North Carolina': 'Yellow', 'Florida': 'Blue', 'Kansas': 'Yellow', 'Louisiana': 'Blue', 'New Mexico': 'Yellow', 'North Dakota': 'Yellow', 'South Carolina': 'Blue', 'Washington': 'Yellow', 'Wisconsin': 'Green', 'Massachusetts': 'Blue', 'Connecticut': 'Yellow', 'Vermont': 'Yellow', 'New Hampshire': 'Green', 'Rhode Island': 'Green', 'Maine': 'Blue', 'Alaska': 'Green', 'Hawaii': 'Green'}
Number of Backtracks: 0
Time Taken (seconds): 0.002511739730834961

Process finished with exit code 0
```

```
Run - isProject3MapColoring
dfs_usa_fwd_chk x dfs_usa_heu x
C:\Users\twins\PycharmProjects\isProject3MapColoring\venv\Scripts\python.exe C:\Users\twins\PycharmProjects\isProject3MapColoring\dfs_usa_heu.py
Color Assignment: {'Alabama': 'Blue', 'Alaska': 'Blue', 'Arizona': 'Blue', 'Arkansas': 'Blue', 'California': 'Green', 'Colorado': 'Blue', 'Connecticut': 'Blue', 'Delaware': 'Blue', 'Florida': 'Green', 'Georgia': 'Yellow', 'Hawaii': 'Blue', 'Idaho': 'Blue', 'Illinois': 'Blue', 'Indiana': 'Green', 'Iowa': 'Green', 'Kansas': 'Green', 'Kentucky': 'Yellow', 'Louisiana': 'Green', 'Maine': 'Blue', 'Maryland': 'Green', 'Massachusetts': 'Green', 'Michigan': 'Blue', 'Minnesota': 'Blue', 'Mississippi': 'Yellow', 'Missouri': 'Red', 'Montana': 'Yellow', 'Nebraska': 'Yellow', 'Nevada': 'Yellow', 'New Hampshire': 'Yellow', 'New Jersey': 'Green', 'New Mexico': 'Green', 'New York': 'Red', 'North Carolina': 'Blue', 'North Dakota': 'Green', 'Ohio': 'Red', 'Oklahoma': 'Yellow', 'Oregon': 'Red', 'Pennsylvania': 'Yellow', 'Rhode Island': 'Yellow', 'South Carolina': 'Green', 'South Dakota': 'Red', 'Tennessee': 'Green', 'Texas': 'Red', 'Utah': 'Red', 'Vermont': 'Blue', 'Virginia': 'Red', 'Washington': 'Green', 'West Virginia': 'Blue', 'Wisconsin': 'Yellow', 'Wyoming': 'Green'}
Number of Backtracks: 2132
Time Taken (seconds): 0.0581212043762207

Process finished with exit code 0
```

5. Results in Tabular format

DFS for US Map:

No of Runs	No of Backtrack	Time(sec)
1	1012451	13.110958814620972
2	1012451	13.13487696647644
3	1012451	14.415603876113892
4	1012451	13.344930648803711

DFS with Forward Checking for US Map:

No of Runs	No of Backtrack	Time(sec)
1	10238	71.71806001663208
2	10238	71.86888551712036
3	10238	72.59776878356934
4	10238	72.32587456211966

DFS WITH SINGLETON AND FORWARD CHECK FOR US MAP:

No of Runs	No of Backtrack	Time(sec)
1	9691	0.30711936950683594
2	9691	0.2932472229003906
3	9691	0.3011901378631592
4	9691	0.30134129524230957

HEURISTICS FOR DFS:

No of Runs	No of Backtrack	Time(sec)
1	701288	48.62122273445129

2	701288	47.30119013786315
3	701288	49.56984423256123
4	701288	48.59866231002135

EXECUTION RESULTS WITH HEURISTICS FORWARD CHECK WITH DFS USA:-

No of Runs	No of Backtrack	Time(sec)
1	1714	0.07276415824890137
2	1714	0.0827786922454834
3	1714	0.07480001449584961
4	1714	0.07632985566584961

Singleton With Heuristic for USA:

No of Runs	No of Backtrack	Time(sec)
1	10286	0.13962650299072266
2	10286	0.1356356143951416
3	10286	0.20747780799865723
4	10286	0.14262890815734863

TABULATION FOR AUSTRALIA:

DFS:

No of Runs	No of Backtrack	Time(sec)
1	0	0.017955541610717773
2	0	0.01798391342163086
3	0	0.019946813583374023
4	0	0.016956806182861328

WITH HEURISTIC:

No of Runs	No of Backtrack	Time(sec)
1	0	0.0013962659072266
2	0	0.0001356356143416
3	0	0.0002074778079986
4	0	0.00014262890815734

DFS WITH FORWARD CHECK:

No of Runs	No of Backtrack	Time(sec)
1	0	0.017955541610717773
2	0	0.01798391342163086
3	0	0.019946813583374023
4	0	0.016956806182861328

WITH HEURISTIC:

No of Runs	No of Backtrack	Time(sec)
1	0	0.000997304916381836
2	0	0.000145326895231456
3	0	0.000123472350002238
4	0	0.000236598462134599

DFS WITH SINGLETON AND FORWARD CHECK:-

No of Runs	No of Backtrack	Time(sec)
1	0	0.000997781753540039
2	0	0.000235000997781753
3	0	0.000123472350002238
4	0	0.00002568153235000

WITH HEURISTIC:

No of Runs	No of Backtrack	Time(sec)
1	0	0.000997304916381836
2	0	0.000145326895231456
3	0	0.000123472350002238
4	0	0.000236598462134599

5. Conclusion

The exploration of the map coloring problem using various computational approaches offers valuable insights into the realm of constraint satisfaction problems (CSPs). Through the implementation of Depth-First Search (DFS), combined with advanced techniques such as forward checking, propagation through singleton domains, and strategic heuristics like Minimum Remaining Values (MRV), Degree Constraint, and Least Constraining Value (LCV), we gain a comprehensive understanding of how complex problems can be approached and solved efficiently.

The key conclusions from this study are:

1. **Heuristic Efficiency:** The application of heuristics significantly enhances the efficiency of solving CSPs. By intelligently narrowing down the search space and prioritizing certain paths over others, these heuristics reduce the computational burden and lead to quicker solutions.
2. **Importance of Forward Checking and Propagation:** Incorporating forward checking and propagation through singleton domains demonstrates a proactive approach in problem-solving. These methods not only streamline the search process by eliminating futile paths early on but also ensure that the solution space is explored more thoroughly and efficiently.
3. **Backtracking Necessity:** Despite advancements in heuristic strategies and forward checking, backtracking remains an essential aspect of solving CSPs. Its role in allowing the algorithm to reconsider and revise decisions ensures that all possible solutions are explored, making it a powerful tool in problem-solving.
4. **Scalability Challenges:** The varying complexity of different maps, such as the intricate USA map versus simpler configurations, underscores the scalability challenges in CSPs. It highlights the need for adaptable and robust algorithms that can handle increasing complexity and constraints without a substantial compromise in performance.
5. **Broader Applicability:** The techniques and strategies developed and applied in the context of the map coloring problem are not confined to this particular scenario. They have broad applicability across a range of CSPs in various domains, showcasing their versatility and significance in the field of computer science and algorithmic problem-solving.
6. **Optimality vs. Performance Trade-off:** This study also brings to light the inherent trade-off between achieving an optimal solution and maintaining computational efficiency. While heuristics and forward checking methods improve performance, they do not always guarantee the most optimal solution, prompting a balance between accuracy and execution speed.

In conclusion, the study of the map coloring problem through these diverse computational lenses offers a rich tapestry of strategies that are instrumental in solving CSPs. It not only deepens our understanding of algorithmic approaches to problem-solving but also provides a framework that can be adapted and applied to a wide range of complex problems in computer science and beyond.