Tugas Individu 3

Map Coloring using Constraint Satisfaction Problem (CSP)

Salsabila Fatma Aripa - 5025211057

Constraint Satisfaction Problems (CSP)

Constraint Satisfaction Problem (CSP) adalah sebuah permasalahan dimana terdapat himpunan variabel yang harus diberikan nilai sedemikian rupa sehingga memenuhi sekumpulan batasan (constraints) yang telah ditentukan sebelumnya.

- State didefinisikan sebagai variabel Xi dengan value dari domain D
- Goal merupakan himpunan batasan (set of constraint) hanya mengijinkan kombinasi nilai untuk subset of variables

Map Coloring using CSP

Salah satu masalah yang dapat diselesaikan dengan CSP adalah **Map Coloring**. Dalam masalah ini kita diminta untuk melakukan :

- Pewarnaan tiap region dengan warna merah, hijau atau biru dimana region yang bertetangga tidak boleh mempunyai warna yang sama.

Implementation

```python
from itertools import product

colors = ["Red", "Blue", "Green", "Yellow"]
states = ["SA", "WA", "NT", "Q", "NSW", "V", "T"]
neighbors = {}
neighbors["WA"] = ["NT", "SA"]
neighbors["NT"] = ["WA", "SA", "Q"]
neighbors["SA"] = ["WA", "NT", "Q", "NSW", "V"]
neighbors["Q"] = ["NT", "SA", "NSW"]
neighbors["NSW"] = ["Q", "SA", "V"]
neighbors["V"] = ["SA", "NSW"]
neighbors["T"] = []


#fungsi get_all_ans bertujuan untuk menghasilkan semua kemungkinan jawaban (solutions)
#yang mungkin dari persoalan map coloring dengan menggabungkan semua kemungkinan kombinasi warna pada tiap wilayah.

def get_all_ans() -> list:
    all_slove = [] #nilai awal berupa list kosong
    slove = {}
    for color in product(colors, repeat=7): #semua kemungkinan warna yg digunakan dalam 7 kota
        for i, c in enumerate(color): #memasukkan hasil kombinasi warna ke dalam variabel slove
            slove[states[i]] = c #mengambil indeks dan nilai warna yang sesuai dengan tiap wilayah
        all_slove.append(slove) #dictionary slove dimasukkan ke dalam list all_slove
        slove = {} #variabel slove dikosongkan kembali untuk digunakan pada kombinasi warna berikutnya
    print(f"There is {len(all_slove)}  sloves in this question") #mencetak jumlah semua kemungkinan jawaban yang dihasilkan
    return all_slove #mengembalikan list dari semua kemungkinan jawaban
```

Implementation

```python
def delete_invalid_ans(all_slove: list): #digunakan untuk menghapus jawaban-jawaban yang tidak valid dari semua kemungkinan jawaban
    is_ans = True
    ans_fit_rule = [] #membuat sebuah list kosong untuk menyimpan jawaban-jawaban yang valid
    for ans in all_slove: #melakukan iterasi melalui setiap state yang ada di dalam jawaban tersebut
        for state in ans.keys():
            for neighbor in neighbors.get(state):
                if ans.get(neighbor) == ans.get(state):
                    is_ans = False #jika warna dari state dan neighbor sama, maka variabel is_ans menjadi False
                    break
            if not is_ans: #jika ada warna yang sama antara suatu state dan salah satu dari tetangganya
                break
        if is_ans: #jika true maka looping selesai dan list ditambahkan kedalam ans_fit_rule
            print(ans)
            ans_fit_rule.append(ans)
        is_ans = True
    if not ans_fit_rule: #jika tidak ada solusi yang valid ditemukan, maka fungsi akan mengembalikan
        return None
    else:
        return ans_fit_rule #jika ditemukan solusi yang valid, maka akanmengembalikn semua solusi yang valid.

def main():
    solutions = delete_invalid_ans(get_all_ans()) #menghapus solusi yang tidak valid dan menghasilkan semua kemungkinan solusi yang valid
    if solutions is None:
        print("No valid solution found.") #tidak ditemukan
    else:
        print(f"Solution: {len(solutions)}") #ditemukan

main()
```

Output

```
There is 2187  sloves in this question
{'SA': 'Red', 'WA': 'Blue', 'NT': 'Green', 'Q': 'Blue', 'NSW': 'Green', 'V': 'Blue', 'T': 'Red'}
{'SA': 'Red', 'WA': 'Blue', 'NT': 'Green', 'Q': 'Blue', 'NSW': 'Green', 'V': 'Blue', 'T': 'Blue'}
{'SA': 'Red', 'WA': 'Blue', 'NT': 'Green', 'Q': 'Blue', 'NSW': 'Green', 'V': 'Blue', 'T': 'Green'}
{'SA': 'Red', 'WA': 'Green', 'NT': 'Blue', 'Q': 'Green', 'NSW': 'Blue', 'V': 'Green', 'T': 'Red'}
{'SA': 'Red', 'WA': 'Green', 'NT': 'Blue', 'Q': 'Green', 'NSW': 'Blue', 'V': 'Green', 'T': 'Blue'}
{'SA': 'Red', 'WA': 'Green', 'NT': 'Blue', 'Q': 'Green', 'NSW': 'Blue', 'V': 'Green', 'T': 'Green'}
{'SA': 'Blue', 'WA': 'Red', 'NT': 'Green', 'Q': 'Red', 'NSW': 'Green', 'V': 'Red', 'T': 'Red'}
{'SA': 'Blue', 'WA': 'Red', 'NT': 'Green', 'Q': 'Red', 'NSW': 'Green', 'V': 'Red', 'T': 'Blue'}
{'SA': 'Blue', 'WA': 'Red', 'NT': 'Green', 'Q': 'Red', 'NSW': 'Green', 'V': 'Red', 'T': 'Green'}
{'SA': 'Blue', 'WA': 'Green', 'NT': 'Red', 'Q': 'Green', 'NSW': 'Red', 'V': 'Green', 'T': 'Red'}
{'SA': 'Blue', 'WA': 'Green', 'NT': 'Red', 'Q': 'Green', 'NSW': 'Red', 'V': 'Green', 'T': 'Blue'}
{'SA': 'Blue', 'WA': 'Green', 'NT': 'Red', 'Q': 'Green', 'NSW': 'Red', 'V': 'Green', 'T': 'Green'}
{'SA': 'Green', 'WA': 'Red', 'NT': 'Blue', 'Q': 'Red', 'NSW': 'Blue', 'V': 'Red', 'T': 'Red'}
{'SA': 'Green', 'WA': 'Red', 'NT': 'Blue', 'Q': 'Red', 'NSW': 'Blue', 'V': 'Red', 'T': 'Blue'}
{'SA': 'Green', 'WA': 'Red', 'NT': 'Blue', 'Q': 'Red', 'NSW': 'Blue', 'V': 'Red', 'T': 'Green'}
{'SA': 'Green', 'WA': 'Blue', 'NT': 'Red', 'Q': 'Blue', 'NSW': 'Red', 'V': 'Blue', 'T': 'Red'}
{'SA': 'Green', 'WA': 'Blue', 'NT': 'Red', 'Q': 'Blue', 'NSW': 'Red', 'V': 'Blue', 'T': 'Blue'}
{'SA': 'Green', 'WA': 'Blue', 'NT': 'Red', 'Q': 'Blue', 'NSW': 'Red', 'V': 'Blue', 'T': 'Green'}
Solution: 18
PS C:\Users\LENOVO\Documents\.UTAMA\SEMESTER 4\6. KB\TUGAS>
```

Thank You ^_^