Exercise 4.1

1.

Constraint satisfaction problem:

Constraint satisfaction problems are mathematical questions defined as as set of objects whose state must satisfy a number of constraints or limitations.

Constraint:

The relation between variables in a state that are considered “legal” or “allowable”. They can involve only a subset of the total of variables. If this relation is not kept, the state is not consistent.

Back-tracking search:

Back-tracking search is a searching technique where the values are chosen for one variable at a time and backtracks when the variable has no values left. It goes backwards.

Back-jumping:

Back-jumping is the process of going backwards to the nearest variable belonging to the conflict set of the failing current variable.

Min-conflicts:

Min-conflicts means to choose the value for a variable that generates the minimum number of conflicts with other variables.

2.

If we start color the state SA in Australia, then we can choose any color of the three. And then the next state is WA where we can choose 2 colors. And there will be possibility chance of choosing only one color that satisfy the problem. Which has possibilities of 3*2 = 6.State T is not connected, it can be mapped with any of the 3 colors. There will be 3 possibilities to color this state. So, there can be 6*3 = 18 solutions that can be obtained by coloring each of the six possibilities with 3 times with 3 colors to the state TA.

There will be 18 solutions for coloring a map with three colors.

Exercise 4.2

1.This problem lists out 10 different states first then it lists out every border created by two neighbor states. The border is represented as a set of 2 neighbor states’ indexes. A color list length of 10 is created. For every border set, the indexes shouldn’t present the same value as color list’s value of those indexes.

The initial state of this problem is the initial fitness which is equal to 0. which assumed the random initialization has no correct arranging.

The goal test is to make state, fitness equal to 17 which checks every border to ensure that every neighbor state doesn’t have the same color.

The successor function is:

```
while color[borders[n].index1] != color[borders[n].index2]
        fitness = fitness + 1
```