Nadya Postolaki

1) [10 points] Given a constraint satisfaction problem with (1) a finite number of variables and (2) a finite domain for each variable

1) how does the number of variables affect the size of the search space?

Increasing the number of variables increases the search space. This is because variables make more for considering and allow for multiple different ways of searching for an answer, thus expanding the search space.

2) how does the size of the domains affect the size of the search space?

The size of domains are directly proportional to the size of the search space. With a larger domain, there are more things to search for and allows for broader searches whereas a smaller domain constrains the search space to the max of that domain, which therefore makes the search space smaller.

Explain your reasoning.

(Reasoning explained within the answers)

2) [20 points] Show in detail how to solve the following crypto-arithmetic problem I + DID = TOO. Write the constraint equations and show the search tree explored by the backtracking algorithm.

```
 DID
+  I
-----
 TOO
```

```
 c2  c1
 D   I   D
+        I
-----------
 T   O   O
```

We know right away that "D" and "I" and "T" cannot be zeros because we never write leading zeros. We see that there are 2 carry overs that occur, the first occuring when I + D giving the output of "O", and we see the same output again with only just "I" so we assume that "D" is the number that's carried over.

Since "I" appears in both "DID" and "I" and the last "D" is added to "I" gives "O" and the next letter in the solution afterwards is also "O", that means that when "D" and "I" are added, there is a carry over "D" to get the same output of "O", implying that "D" has the value of 1. Again "I" and "D" were added together therefore another carry over occurred and the next addition performed is D + D, which produces a number less than 10 which equals "T".

Since we add "D" and "D" together to get "T", and we know that D = 1, we can assume that T = 2.

D = 1
T = 2
I = ?
O = ?

So now the equation looks like this:
 1I1
+ I

 2OO

Now we continue with the knowlege that since we carried over "D" (1), the only number less than 10 that when added by 1 turns into a double digit number is 9, therefore I = 9. Since I = 9 and D = 1, I + D = 9 + 1 = 10, and since D = 1 and we carry over the 1 to the next integer, we know that O = 0.

D = 1
T = 2
I = 9
O = 0

So testing the logic, the equation now looks like this:
 191
+ 9

 200

Since 9 + 191 = 200 is true, we know the solution mentioned above is correct for I + DID = TOO.

3) [20 points] This question is on the arc consistency AC-3 algorithm
Show how the arc consistency AC-3 algorithm is able to detect the inconsistency of the partial assignment {Wa=red,V=blue} in the map of Australia.

AC-3 quits the loop as soon as an inconsistency is detected. As soon as the partial assignment occurs with WA = red and V = blue, AC-3 sees the shared constraint of SA between WA and V and colors that portion green. Then NT shares constraints with WA and SA, as well as NSW sharing with SA and V, would be colored blue and red respectively. We then have Q left, but all 3 colors are adjacent to it, therefore it is inconsistent and cannot be completed with the partial constraint. Same goes if we back tracked and went through all possible color combinations for Q, NSW and NT would be inconsistent depending which territory is colored first, thus the map needn't be completed.

Try the same AC-3 algorithm to detect the inconsistency of the partial assignment {WA=red,NSW=red}. Does the algorithm detect the inconsistency or not?

The algorithm doesn't detect the inconsistency of the partial assignment because WA and NSW does not change the constraints of the surrounding territories and they all have the same constraints, therefore the algorithm cannot further narrow down possiblities.

If not, why AC-3 is not always capable of detecting inconsistencies?

When all surrounding territories have the same constraints, they are unaffected by the already set partial constraints so the algorithm is unable to narrow down possibilities of other territories.

4) [10 points] How many solutions are there for the map-coloring problem for the Australia map (textbook Figure 6.1) when using three colors? Explain your results.

18 total solutions. When you completely color the map once, you can rotate the colors around which gives you 3 total possibilities, then theres the ability to color T any of the 3 colors as it is not constrained by anything on the map, so thats 3 x 3 = 9 possibilities in the first variation colored. Then we switch the colors of SA and NT, leaving WA as it originally was in the first coloration of the map, and again the colors can be rotated which allows for 3 solutions and again T allows for 3 x 3 = 9. We then add 9 + 9 for a total of 18 solutions.

5) Extra credit [20 points] Use the AC-3 algorithm to label the following figure. Explain step by step how the algorithm works.

See below image for solving

AC-3 algorithm won't be able to completely label the figure for T's L and N and fork M as they have two different viable solutions, of which the outer contraints make no difference in defining. There are two different possible solutions due to the lack of constraints on the inner angles L, M, and N.