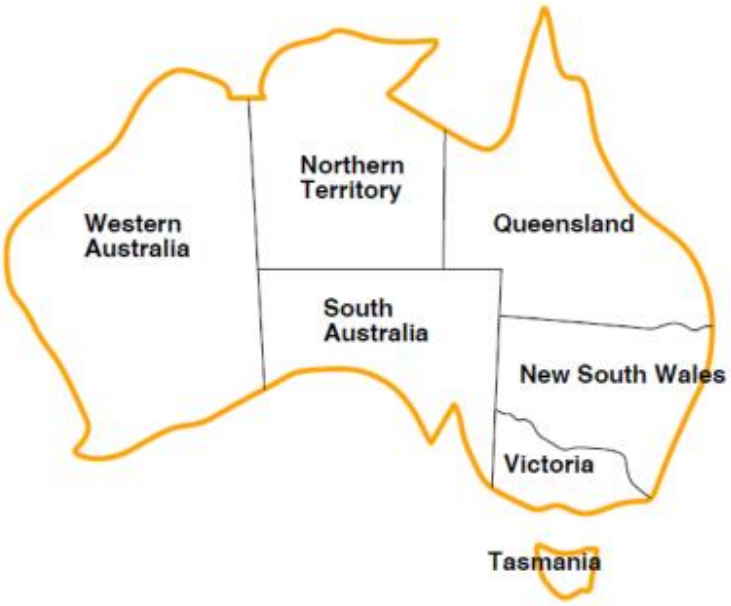
10/23/2014 9:16 AM

CSCE 420
Programing Assignment #5
due: Sat, Nov 1 (by midnight, i.e. end of Saturday)

Objective

The goal of this problem is to implement an algorithm in C++ for solving Constraint Satisfaction Problems. This is in *preparation for the next assignment*, where you will be using it to solve a multi-agent coordination problem. But for simplicity, for this project, the application will be to solve the **map-coloring problem for Australia**, as described in the textbook.



Implementation

The main goal of this assignment is to write the **backtracking algorithm** for CSPs in C++. Follow the pseudo-code in the textbook (Figure 6.5, page 215). The algorithm is fairly straightforward, recursively picking a value for an unassigned variable and then testing to see if any constraints are violated.

A critical part of this assignment will be writing the function that tests whether a partial variable assignment violates any of the constraints, such as "bool satisfied(constraints,assignment)". Your satisfied() function will have to handle and test all the types of constraints you use. For map coloring, all the constraints are inequalities, such as "neq WA NT". For other CSP problems (as we will see in the next project), you might have to extend this function to evaluate other kinds of constraints (but for now, keep it simple). Importantly, you might not be able to evaluate all of the constraints given a partial assignment if the variables involved in a constraint are not yet defined in the assignment.

Your CSP program should read in a problem specification file that gives the variables, domains, and constraints that encode the problem. Your program should have a pre-processing phase to read in the problems description and set up the internal data structures before invoking the backtracking search. To illustrate the format, here is the input file that gives the specification of the problem by defining the vars, domains for the Australia map-coloring problem:

```
# vars
WA NSW TA V NT QL SA
# domains
WA red green blue
NSW red green blue
TA red green blue
V red green blue
NT red green blue
QL red green blue
SA red green blue
# constraints
neq WA NT
neq WA SA
neq NT SA
neq NT QL
neq SA QL
neq SA NSW
neq SA V
neq QL NSW
neq NSW V
```

Minimum-Remaining Values (MRV) Heuristic

Note that if the regions of Australia are given in the (sub-optimal) order shown above, the CSP search is rather inefficient, and a great deal of backtracking occurs (though it should still find a solution). It is unfair to "give" the variables in the optimal order, since you general do not or do not have control over this for most problems. Thus, you should incorporate the MRV heuristic into your search algorithm to make more intelligent choices about the order of variables to work on. As described in the book, MRV does this dynamically by keeping track of domain size. It requires a minor modification of the same search routine where, instead of always choosing the next variable in the initial order, you choose the variable with the least remaining values. (the specific line in the pseudo-code is: `var ← Select-Unassigned-Variable(csp)`). This might involve testing all the remaining values for all the unassigned variables *each time* you recursively call the search algorithm to determine which values will violate a constraint (given the partial assignment at that stage), and discarding those values from the domain of that variable.

Without the MRV heuristic, backtracking takes 42 iterations to solve the map-coloring problem. With MRV, it only takes 8.

What to Turn in

- You will submit your code for testing using the web-based CSCE *turnin* facility, which is described here: https://wiki.cse.tamu.edu/index.php/Turning_in_Assignments_on_CSNet
- Include a brief document that describes how to compile and run your code.
- Include a transcript showing the solving the Australia problem **with** and **without MRV**, using the variable order shown above. Print out tracing information with each step, showing information such as the partial assignment, which variables and values are selected, and when backtracking occurs. (For the longer trace, you may abbreviate the output by chopping out the middle.) Make sure the output shows the **total number of iterations** made during the search and the **final solution** (variable assignment).