CSC520 Spring 2019 – Assignment 2

Due by midnight of **Feb 15th 2018**.

This assignment includes both conceptual and code questions. It must be completed individually. You may not collaborate with other students, share code, or exchange partial answers. Questions involving answers or code must be emailed to the instructor or TAs directly or discussed during office hours. Your answers to the conceptual questions must be uploaded to Moodle as a pdf file titled "Assign2_UnityID.pdf". Your sourcecode must be submitted as a self-contained zip. All code must be clear, readable, and well-commented. Note: The code will be tested on the NCSU VCL system("CSC520_VCL"). You are advised to test your code there before submission.

Question 1 (10 pts)

Use the lexicon:
n – A student can log in the Mypack website.
r – A student can go inside Hunt Library .
i – A student has an NCSU ID card with him/her.
g – The students will get a NCSU ID card
q – The NCSU student.
oc – A student who lives on campus.
ic – A student who lives off campus
ad – A student can go inside dorms
p – Peter is a student .

- For all the students of NCSU, they get a NCSU Student ID card.
- All the students from NCSU can log in the Mypack website.
- NCSU Students can go inside Hunt Library only if they have am NCSU ID card with them.
- All of the NCSU students live on or off campus.
- The students who live on campus can enter dorms only if they take their NCSU ID cards with them, while others who live off campus cannot.
- Peter is an NCSU student.
- Peter lives on campus. But he cannot go inside the dorms and Hunt Library as he has lost his card.
- Peter cannot log in the Mypack website.

a. (5 pts) Use propositional logic to determine if the specification is consistent. If the sentences are not consistent, use resolution to derive a contradiction. If they are consistent, use the truth-table method to show at least one model.

b. (5 pts) What if anything changes if Peter can log in the Mypack website? Be specific in showing what's different.

Question 2 (50 pts)

You have been given three text files that describe binary constraint problems. The first line of each file is `VARS` following that line is a list of variable names followed by their respective values. Variable names are represented by sequences of characters and are separated from values by a : Values are represented by integers. The list will end with a single line stating `ENDVARS`. Following the variable declarations there will be a set of constraints starting with the line `CONSTRAINTS` and ending with the line `ENDCONSTRAINTS` Constraints are represented line by line using prefix notation. The first symbols in each line represents the constraint function which must hold over the arguments. Following the constraint symbol there will be a sequence of variable names *or values* separated by whitespace. The constraints apply to all of the variables listed. *NOTE: Apart from != and = most will be binary constraints.*

- `!=` Not equal or Alldiff
- `=` equal or Allmatch
- `<` value less than.
- `>` value greater than.
- `<1` value one less than.
- `>1` value one greater than.
- `<>1` value one less or one greater than.

The problems are described briefly here:

- **Map Coloring:** You have been given two maps. One `Map_v.txt` which represents a partial map of southeastern states of USA consists of Alabama, Mississippi, Georgia, Tennessee, Florida. The variables are initials of each state A, M, G, T, F. Neighboring states should have different colors and specified in the constraint file. Allowable colors are numbered as $\{1, 2, 3\}$ Initially all states have the domain of $\{1, 2, 3\}$ The other of which represents the contiguous 48 US states and expands the range of values to 5.

- **Sudoku:** A Sudoku puzzle has 81 variables, one for each square. The variable names follow the convention of textbook Figure 6.4. Rows are numbered from A1 through A9 for the top row (left to right), down to I1 through I9 for the bottom row. The empty squares have the domain $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and the prefilled squares have a domain consisting of a single value. In addition, there are 27 different constraints defined in the constraint file.

  Class Assignment: You have to assign classes for N subjects, $S_1, S_2, , S_N$ to k Professors. A Professor can teach one or multiple classes. The domain of the variable S_i, $i \in \{1, N\}$ specifies the Professors who can teach the subject S_i. The constraint are specified in $S_i \neq S_j$ format, indicating that these two courses have overlapping time slots and Professors.

  Zebras: As described in class you have five houses with an owner, a pet, a drink, a cigarette, and a color of each. We will do assignment over each house.

Your task in this assignment is to implement two solver algorithms for the constraint problems: BACKTRACKING, and MIN-CONFLICTS as described in the lecture and textbook. Both algorithms should be implemented as part of a single package that takes as input a problem file and an argument specifying the algorithm to use and as output returns a file specifying a complete and satisfactory assignment of values to the variables *or* a final assignment with conflicts noted if no solution is found. You should use a max steps parameter of 20. When run with MIN-CONFLICTS your code must print out the number of steps taken to find the assignment. When run with backtracking search your code must print out the number of backtrack steps taken. As always your code must be clear, readable and well documented. For this assignment you are permitted to use third party libraries for graph data structures as necessary. In your report you should compare and contrast your algorithms and problems. Which approach worked better for each of the problems and how could they be improved.